**Notizen zu: SECOMO – Cost Model for estimating eco-costs of software-driven systems**

Abstract:

* Concepts: for description of ecological costs
* Models: auxiliary(?) models expressing ecological costs & circumstances causing them
* 🡪 Algorithmic eco-cost estimation models (taking auxiliary models as input)

Goal: Show impact of development decisions IN EARLY stages! (based on information available)

* Approach (Secomo) of the thesis = fulcrum (?) for green software engineering methods, estimate eco-costs before implementation has started

Übersicht:

Foundations:

* Green ICT
* Estimation Techniques in Software Engineering
* (Example: Shopping Kart example specification - KobrA)

Expressing Eco-Costs with SECoMo

* Core Concepts
* Eco-Cost Metrics
* Auxiliary Models

Estimating Eco-Costs with SECoMo

* Basic Concepts: Estimation Techniques
* Early Estimation of Eco-Costs
* Estimating Eco-Costs on the Intermediate Level of Detail
* Estimating Eco-Costs on the Advanced Level

Using SECoMo

* Leveraging SECoMo in Software Engineering
* Evaluation

# Introduction

* Software in our lives
* Software & Global warming / greenhouse gas emissions / Co2 emissions -> software plays a role
* 2 strategies
  + Minimize amount of energy consumed by ICT
  + Increase amount of renewable energy used by ICT
* Video on Demand example – align energy consumption w/ green energy production
* Efforts to counteract
  + Reduce energy of data centers (bas10), (ber12), (zel14), …
  + Projects about better alignment of consumption & production, e.g. about
    - Virtualization -> enable use of unused hw(?)
    - Cooling systems (data centers)
    - End user products / HW 🡪 increased energy efficiency
* BUT – software never had the same big focus before – just now starting 🡪 Green / Sustainable Software Engineering; ICSE / IEEE Software – now deal with it
* Aspects
  + Life Cycle / Process models with SUSTAINABILITY AWARENESS
  + Analysis: which effect do design patterns, programming languages, code refactorings have (Nau11, BS13, Nou12a, SPC14) on ENERGY CONSUMPTION caused by SW
  + Measurement / Estimation tools (WGR13, Kon08, Seo09a) (JouleUnit)
  + Sustainability Metrics (Joh12, MP15)
* Why not more / further evaluated, sophisticated solutions? 🡪 EXPENSIVE! (late in the process, change means costs!)  
  “most of the approaches developed so far focus on optimizing the software during the implementation phase” – better would be possibilities to do it EARLIER! Already in design phase / requirements, etc. – but that requires a way to estimate it 🡪 “analyzing potential optimization early in the sw development process”   
  - some models exist – e.g. greensoft or other lifecycle models   
  – but lack in concrete methods to estimate actual costs to have hard facts

- + NEED to create awareness in the earliest stages!  
- most methods add further complexity with additional documents, not integrate it in sw development process (or not so much)

* 🡪 SECOMO provides a method for that – **Software Eco-Costs estimation model**
* Research Challenges
  + **Unknown ecological impact of development decisions** – hard to say (yet) which impact certain decisions have, thus it’s hard to consider trade-offs to reach higher sustainability / energy efficiency
  + **Inability to define and communicate reasonable eco-goals for programmers** – lack of methods / models that are appropriately integrated
  + **Lack of information** – of what influences sustainability / eco efficiency, early in the sw dev process, hard to define SLA’s etc.
  + **Lack of tailor made metrics** – many metrics suggested; but often very generic, or not fitting for early software development phases 🡪 better: to have metrics expressed in sw engineering concepts (e.g. specifications)
* Research Goals
  + Estimation of Eco-Costs from Different Viewpoints -> C1, C2
  + Better Eco-Cost Metrics -> C4
  + Better Communication of Goals, Trade-Offs and Ecological Impacts ->C1, C2
  + Adaptability (of estimation approach) -> C3

**SECoMo – Software Eco-Cost Model**

* Software Eco-Cost ESTIMATION model, approach for / supporting sustainable software engineering!
* Auxiliary Models (“Hilfs-“modelle) 🡪 enable the representation of eco costs + causes!! 🡪 for a clear communication!
  + Enhance traditional sw specifications (KobrA OR A SIMILAR specification technique?!) with eco-costs
  + Other models can be derived from them (Green Specifications)
  + …
* Estimation Models 🡪 mathematical estimation model: costs of a software system from several perspectives and in several (early) stages
  + ESTIMATE eco-costs that will actually arise when a SW is executed
  + “all levels of granularity” – whole system down to single components, e.g. data
  + Worst-, Best-, Average eco-costs for
    - All HW platforms (?)
    - Different user types (?)
    - Different points in time
  + 3 levels of detail
* Basis: Eco-Cost Metrics 🡪 describe eco-costs in a unified way
  + 11 new metrics
* 🡪 SECoMo allows developers to measure/estimate/realize impact of their design decisions early + enable (cheap) changes, inform them + support them to make “well informed choices”

# Foundations

## GREEN IT

Green ICT

* Starting point: Study by Gartner (Pet07) 🡪 ICT is “a MAJOR contributor” to GHG emissions / global warming, 0.5 % of worldwide emissions in 2007 🡪 thus eco-efficiency of IT MUST be improved
* GREEN IT.. “this will help explain the context of this thesis in relation to other software related approaches and clarify how they can enhance each other” ?!
* (MP15) Green-by-IT and Green-in-IT
  + Green by IT: use IT to reduce greenhouse gas emissions (e.g. use video conferencing instead of face-to-face meetings -> travel)
  + **Green in IT: Increase energy efficiency** of IT artifacts itself (HW, SW) **(focus here)**
* **Optimization** **of data centers**
  + Better infrastructure -> all hardware elements improved in energy consumption
  + PUE metric – how much of DC energy consumption is caused by IT? (AVE+12) 🡪 goal: PUE of 1 = all energy is consumed by IT, i.e. not other cause (like ventilation)  
    🡪 not perfect, not totally accurate, BUT: helps to raise awareness !
  + Best practices: “free cooling systems”, ice storage facilities (to store energy?), solar panels, higher average room temperature (cold not necessesarily needed) …
  + CUE (Carbon Usage Effectiveness) (Aze+10) – how much of the Carbon emissions caused by the DC is due to IT energy consumption
* 🡪 INCLUDE impact of HVAC systems in software eco-cost estimation!
* **Optimization of ENERGY need**(?) of HW – servers
  + E.g. improvement by HP (Hew16); high efficiencies CAN be reached (80 Plus Titanium)
  + Causes: each component could be responsible
    - PSU (power supply unit)
    - CPU
    - Hard disks
  + Approaches to increase CPU efficiency
    - New processor designs
    - Dynamic frequency scaling
    - Disable unneeded cores
* **Optimization of ENERGY need**(?) of HW – components in other electronic devices (laptop, smartphone, tablets) + TV’s, Game Consoles, …
* **DISADVANTAGE:** 
  + Improved HW means NEW hardware -> issue of IT disposal, waste!!!
* 🡪 another important aspect: improve EXISTING HW, leverage it!
  + Virtualization – share resources & allow better alignment, use of unused resources, … 🡪 reduces IDLE energy consumption as LESS HW is needed overall
  + E.g. FIT4GREEN project (Bas+10), All4Green (Bas+13), DC4Cities (Kli+15)
  + Other approaches: Frequency scaling, Dynamic voltage scaling, Standby modes
* 🡪 INCLUDE consideration of these approaches in software eco-cost estimation!
* Optimization by: using SLA’s
  + Sla’s grant the customer a specific level / quality of a service (source?)
  + GREEN SLA’s enhance SLA’s concerning more environmental issues
    - E.g. define LOWER availabibiliy in the night -> allow for better options to increase energy efficiency
  + CLEAR STATEMENTS needed!

Green Software

* Many approaches developed – can be combined with SECoMo!!
* Overview
  + GREENSOFT – reference model, all lifecycle phases  
    🡪 development process enhanced by agile sustainability process (reviews etc.)  
    🡪 create awareness, measure(?) it, communicate it with all stakeholders  
    NO CONCRETE metrics, methods, models about ecocosts…
  + Many models about IMPLEMENTATION phase
    - SEEDS – energy-optimiuation decision support (MPC14)  
      🡪 code optimization to achieve energy consumption improvement
    - General: code optimization not focused on energy efficiency as it is hard to grasp
  + Influence of Software design patterns on energy consumption
    - Lit05, Sah12, BS13
    - Yes, they can have a huge influence
  + Influence of Code Refactorings
    - SPC14
    - (?)
  + Effect of Programming Languages on energy usage
    - Nou12b
    - E.g. pearl high; C, C++, Java,… low
  + Effect of certain components, algorithms, browsers, (von mir: dev. Environments!),
  + Energy MEASUREMENT in general – PREREQUISITE
    - Can be by direct use of HW, by estimation or by simulation
    - GreenTracker
    - Bun09;
    - LEAP framework (MPC14)
    - PowerAPI (estimation) (Nou12b)
    - JouleUnit (WGR13)
  + REQUIREMENTS phase
    - How to better consider sustainability concerns? Pen16   
      🡪 EnvironSiSE and AMDiRE approach for relevant artifacts
  + Green Specifications
    - Eco-cost parameters as part of software specifications
    - Can support SLA’s
    - 🡪 HOW TO OBTAIN the eco-cost information?
* OR – quite a different thing
  + Measuring social & economic impacts of green software   
    MP15, ASK14, Sah14

Green Metrics

* Literature survey: Morage, Bertoa (81 measures?!)
* Example Energy Efficency Measures
  + PUE Gro13
  + Power Consumption Hin12a, Hin12b, KLL12, Nou1b
  + Useful work done Joh12
  + Energy Costs at different levels SMM08b, Seo09b
* Resource Optimization measures
  + Response times (mar12), (kip11b)
  + Availability (Kip12)
  + Learnability (alb10)
* Capacity
  + Workload (goi13)
  + Percentage of used functionality (LFF13)
  + Used functionality fixed to relative energy ratio (Gro13)
* Perdurability
  + Defect density (Alb10)
  + Reliability (Kip12) (Kip11b)
  + Modification requests (Sea03)
* Literature review Bozelli et al (BGL13)
  + 66 metrics for energy consumption of software
    - Energy
    - Performance
    - Utilization
    - Economics (dollar)
    - Performance/energy
    - Pollution (Co2 units)
* Metrics either too abstract or too detailed, too specific for one problem :/

## Estimation Techniques in Software Engineering

Development Cost and Effort Estimation

# Using SECoMo

## Leveraging SECoMo in Software Engineering

Pro’s and Con’s of using SECoMo – Costs and Benefits

* Usually: saved ecological costs outweigh monetary costs, but not always
  + Costs of creating auxiliary models – overhead, but not if connected to a process which already creates the models
* When is SECoMo beneficial?
  + When ecological friendliness is an important aspect in the SW system usage environment
  + When energy consumption should be minimized
  + When local energy sources should be used – as it provides information about the actual energy consumption
  + When new HW should be purchased to minimized the overall costs of usage – with the knowledge about the resource usage, optimal hw can be purchased

Using SECoMo in Existing Software Engineering Processes

… in **Traditional Approaches** (e.g. Waterfall, **V-Model**) – FIGURE 12.2!!

* SECoMo can be used throughout the whole development cycle!!
* Requirements Engineering
  + Ecological costs = key concern for formulating requirements
  + Easier to make decisions as more information about ecological costs are available
  + Good: supports early decisions about sustainability -> less actual costs (than changes later, e.g. after implementation & then measured)
* Design Phase
  + Adaptability of model 🡪 estimations can be redefined & improved
  + Design decisions can take eco-cost estimations into considerations
  + E.G. Architecture design – possible to find a solution that is more eco-efficient
* Implementation Phase
  + 🡪 Eco-cost constraints can act as Non-Functional Requirements for development
  + Possibility to define specific or general eco-cost requirements that developers have to adhere to
  + E.g. in combination with green specifications / SLA’s
* Verification / Validation
  + Use auxiliary models to validate the SW against! Does the software adhere to the eco-cost requirements 🡪 necessary to MEASURE ecological costs precisely
  + Analyze reasons why it does not fit
    - Requirement too strict?
    - Not implemented well enough?
  + + update the ecocost models with actual values 🡪 improve accuracy of future estimates
* Maintenance
  + SECoMo 🡪 helps to analyze situations which were not foreseen
  + Update auxiliary models -> to adapt models to support future projects
  + Helps to support change situations 🡪 calculate new estimates! To find optimal solution

… in **Agile Approaches (e.g. SCRUM)**

* Mainly 🡪 early-level estimates relevant, less the formal documents!!
* Backlog Creation / Requirements Engineering
  + Must Have vs. Nice To Have 🡪 nice to haves can be regarded relating to eco-costs 🡪 remove them if too high!
  + Perform estimates for NiceToHaves
  + Optimize Must Haves for Eco-Costs
  + Add new eco-cost requirements
* Sprints
  + Keeping track of ecological costs – estimated vs. actually measured(?)
  + At the end: validate and analyze if successful, if not – reasons?
* Estimations
  + Less relevant, as rapid implementation is the focus..
  + But if not regarded 🡪 possibly wasted time -> overhead
* Goal: LOW OVERHEAD – vs. high overhead of specific estimates..
  + Prefer EARLY level estimates 🡪 (why?)
  + Documentation still important
* End of Sprint
  + Use Secomo to create new estimate / update existing ones
  + + at the very end 🡪 auxiliary models provide a very good documentation + measured ecological costs are already included

… COMBINING SECOMO WITH EXISTING SUSTAINABLE SW ENGINEERING APPROACHES!!!

* GREENSOFT -> analyze in more detail / example of how to fit?
  + SECoMo fits into the general frame of the model
  + Relates to the development / usage phase
  + Related to the effects (first, second, third-order) – how it can effect sustainability issues
    - Reducing hw and resource usage
    - Facilitate use of renewable energy
  + OFFERS NEW SUSTAINABILITY metrics
  + + adds recommendations
* Requirements Engineering
  + Integrate with Penzenstadler Approach (MP15) – sustainability goals & rules
  + Information of the requirements -> important input for SECoMo
  + Quality requirements 🡪 can use metrics of SECoMo
  + (integrate with others)
* Help for architectural / design decisions?
* Specification
  + Of data types, functions, etc.
  + Optimize specifications
* Development
  + Eco-cost requirements provided
  + Approaches (from chapter 2?)
  + Choice of development language – how does SECOMO fit with languages like eco?
  + Choice of framework (e.g. SEEDS) – to support code optimizations – other frameworks?
* Validate fulfillment of requirements
  + With existing tools, e.g. JouleUnit
  + (others?)

Ideas for my seminar thesis

(Explore eco-costs for application features AND architectures -> architecture foundation for seminar thesis, too (& compare) – can be used to define REQURIEMENTS! (also requirements part can be in there / compared)