Sustainable Computing: Informatics and Systems 1 (2011) 294–304

Contents lists available at ScienceDirect

Sustainable Computing: Informatics and Systems

journal homepage: www.elsevier.com/locate/suscom

The GREENSOFT Model: A reference model for green and sustainable software and its engineering

Stefan Naumann*, Markus Dick, Eva Kern, Timo Johann

Trier University of Applied Sciences, Umwelt-Campus Birkenfeld (Environmental Campus Birkenfeld), ISS - Institute for Software Systems, P.O. Box 1380, 55761 Birkenfeld, Germany[1,2,3]

ARTICLE INFO

Article history:
Received 6 May 2011
Accepted 21 June 2011

Keywords:
Green Software
Green IT
Green by IT
Software Engineering
GREENSOFT Reference Model

ABSTRACT

The resource and power consumption of ICT is still increasing, but also the benefits of ICT, e.g. in finding more efficient solutions for environmental problems. To date, it is not clear, whether the resource and energy savings through ICT overbalance the resource and energy consumption by ICT, or not. Up to now, manifold efforts of Green IT address the environmental aspects of sustainability considering computer hardware. However, there is still a lack of models, descriptions or realizations in the area of computer software and software process models. In our contribution, we first propose definitions of the terms "Green and Sustainable Software" and "Green and Sustainable Software Engineering", then we outline a conceptual reference model, the GREENSOFT Model. This model includes a cradle-to-grave product life cycle model for software products, sustainability metrics and criteria for software, software engineering extensions for sustainably sound software design and development, as well as appropriate guidance.

1. Introduction

Green house gas effects, climate change and, as a means to mitigate these, Sustainable Development (abbr. SD) are the major challenges, mankind is faced with in the world of today [1]. Information and Communication Technology (abbr. ICT) makes up a considerable constituent of these complex challenges. On the one hand, ICT has the potential to push SD, if it is used to optimize material flows or to substitute material products with their virtual counterparts, which reduces energy and resource consumption [2]. On the other hand, its ever-increasing usage induces rising demands for energy and resources [3]. As an effect, the approximated energy consumption of U.S. data centers increased from 28 billion kWh in 2000 up to 61 billion kWh in 2006 [4]. Meanwhile, estimations rose from 58 billion kWh in 2000 up to 123 billion kWh in 2005 on a global scale [5].

Previous academic research discussed the relationship between SD and ICT. These works focus on the impacts of ICT on environmental sustainability [6] or on the balance between energy and resource savings by ICT and energy and resource consumption of ICT [7]. Unfortunately, up to now, no consent has been found on whether energy and resource savings by ICT will exceed its energy and resource consumption or not.

Our contribution presents the GREENSOFT Model, a model of "Green and Sustainable Software" addressing both challenges: the reduction of the energy and resource consumption in ICT, as well as the use of ICT to contribute to SD. In general, the proposed reference model can be classified into the new research field of *Sustainability Informatics* [8].

2. Background and related work

Until now, there are many publications available discussing the relationship between ICT and SD. Berkhout and Hertin [9] identified three main impacts of ICT on the environment, while summarizing literature on the topic. These impacts are: first-, second-, and third-order impacts. First-order impacts are environmental effects that result from production and use of ICT, i.e. resource use and pollution from mining, hardware production, power consumption during usage, and disposal of electronic equipment waste. Second-order impacts are effects that result indirectly from using ICT, like energy and resource conservation by process optimization (dematerialization effects), or resource conservation by substitution of material products with their immaterial counterparts (substitution effects). Third-order impacts are long term indirect effects on the environment that result from ICT usage, like changing life styles that promote faster economic growth and, at worst, outweigh the formerly achieved savings (rebound effects). These effects do not appear sequentially and disconnected. In reality they are nested, which means that second-order effects can only emerge on the

* Corresponding author. Tel.: +49 6782 17 1217; fax: +49 6782 17 1268.
E-mail addresses: s.naumann@umwelt-campus.de (S. Naumann), m.dick@umwelt-campus.de (M. Dick), e.kern@umwelt-campus.de (E. Kern), t.johann@umwelt-campus.de (T. Johann).

[1] http://www.umwelt-campus.de/.
[2] http://iss.umwelt-campus.de/.
[3] http://www.green-software-engineering.de/.

2210-5379/$ – see front matter © 2011 Elsevier Inc. All rights reserved.
doi:10.1016/j.suscom.2011.06.004

basis of first-order effects and third-order effects can only appear as ramifications of second-order effects.

These findings, which focus mainly on environmental aspects, can be refined to address also human and social sustainability issues of ICT production and use [6]. Here, the effects of ICTs are called "effects of ICT supply" (first-order effects), "effects of ICT usage" (second-order effects), and "systemic effects of ICT" (third-order effects). They substantially describe the same effects as Berkhout and Hertin described, but without limiting these to environmental issues.

Hilty [10] developed a model that combines standard Life Cycle Assessment (abbr. LCA,) [11,12] with the afore mentioned effects, to show the potential impacts of ICTs on the life cycle of other products. He applied his model to fields of ICT applications, which have been said to have a high carbon dioxide reduction potential and he identifies obstacles that hinder the use of these potentials.

Fuchs [13] discusses the relationship of ICT and SD and dismantles the myths that teleworking has reduced the need to travel and that information economy is nearly weightless and dematerialized. Hilty et al. [14] showed that new versions of a software product on current hardware are not necessarily more productive than the older version and can be even less productive.

Behrend et al. [15] investigated the problem of conflicts that are exacerbated by the extraction of raw materials widely used in electronic products and components. Related to social sustainability, these conflicts can be valued as social sustainability related third-order effects of ICT production. First- and second-order impacts on social and human acceptability of ICT production and electronic waste disposal have been investigated by Borman and Plank [16], Chan and Ho [17], Prakash and Manhart [18], and many others [19–22]. Their discussions range from wages below the statutory minimum to excessive working hours and from forced overtime to health and safety risks.[4]

Coroama and Hilty [7] investigated the two popular directions of ICT: reducing the energy consumption of ICT itself and reducing the environmental impacts on other sectors by ICT. They make the case that these two directions should be integrated in order to assess net green house gas emissions in specific application areas of ICT. They argue that reports in this area (e.g. [23]) independently look at these two aspects without taking into account that energy savings in ICT application areas may be overcompensated by the growth of ICT appliances that are necessary to achieve these savings.

Mocigemba [24] introduced a sustainable computing concept that can be used to classify and understand the different directions in this area. This concept implements three different foci at a time for the material level of ICT (hardware) and for the informational level of ICT (software). The first level focuses on the product, the second on the production process, and the third on the consumption process. The given examples mainly deal with social and human issues of computer use. They hardly consider neither environmental issues nor that energy and resource consumption of ICT is related to the potential of energy and resource conservation induced by ICT.

Abenius [25] relates effects of ICT products to their life cycle phases and identifies that "Green Software" can mitigate first-order impacts of ICT usage. She proposes a grouping of "Green Software" into "existing tools" and "new inventions", which are then further divided into the categories "monitor and measure" as well as "increasing performance".

Albertao et al. [26,27] relate common software quality aspects like modifiability, portability, or performance to the interdependent areas of SD, namely economy, society, and environment. They also present some metrics, usually used to measure these quality aspects, but lack showing how the resulting values should be interpreted in order to achieve the intended effects on SD. Besides these, they propose a simple improvement cycle that can be applied in software development projects and that has the objective to improve the sustainability of the software product from one release to the following.

Arndt et al. [28] discuss implications of evolving releases of a widely used text processor and relate these to Green IT and SD. As a solution to cope with sustainability issues during software design and development, they propose the so called "Grand Management Information Design", which tries to transfer the Bauhaus design principles to immaterial software products. Dick and Naumann [29] presented a generic enhancement for software development processes that institutionalizes the consideration of sustainability issues during software design and development.

Kansal et al. [30] introduced "Joulemeter", a tool that estimates the pro rata power consumption of virtualized servers running on one hardware server. It uses a power model that leverages CPU usage and disk IO of each virtual server in order to estimate its power consumption. The specific power parameters of server hardware are determined with power meters (either an internal or external one), whereas the model parameters are learned in situ. Zapico and Turpeinen [31] introduced "Greenanalytics", a tool that visualizes the impact of websites on the environment by using data from Google Analytics. Amsel and Tomlinson [32] presented "Green Tracker", a tool that estimates the energy consumption of software in order to help users to make informed decisions about the software they use. Both tools have the objective to raise awareness of software induced energy consumption among users as well as software developers. Naumann et al. [33] presented the "Power Indicator" (now called "Green Power Indicator"), an add-on for a popular web browser, which visualizes, whether or not the web server that hosts the current websites powered by renewable energy.

Capra et al. [34] investigated the impacts of different software systems on IT energy consumption. They state that energy efficiency issues are mainly focusing on hardware. However, in the rare cases in which they focus on software, it usually is on embedded software, where energy is a strictly limited resource. They find that different software products that satisfy the same functional requirements differ significantly in their direct energy consumption. Furthermore, they showed that improving time performance of a software product may not necessarily lead to lower energy consumption.

There are also some supporting tools and guidelines available for software engineers that present best practices on how energy efficiency of software can be optimized, e.g. [35], on how data transfer volumes of websites can be reduced, e.g. [36–38], or on how software artifacts can be instrumentalized and analyzed in order to measure the energy consumption induced by them [39,40].

3. What is Green and Sustainable Software?

Before presenting our model, it is necessary to clarify what we understand by "Green and Sustainable Software" and "Green and Sustainable Software Engineering". Hence, we give two definitions in this section. These definitions are based on the background of holistic product life cycles in the sense of LCA or a "cradle-to-grave" approach, the findings on the three different levels of impacts of ICTs on SD, and Hilty's work on impacts of services offered by ICTs on the life cycles of other products and services.

In principle, a software product that is attributed as "Green and Sustainable" should itself be as sustainable as possible. This means that economic, societal, and ecological impacts, as well as impacts on human beings that result from the product over its

[4] Many more publications investigating these and related issues can be easily found here: http://ewasteguide.info/biblio_archive.

whole life cycle, should be as small as possible. Most obvious are the first-order effects (or: effects of ICT supply), like performance requirements, network bandwidth, hardware requirements, and product packaging that directly lead to a more or less specific demand of energy or natural resources. The second-order effects (or: effects of ICT usage) evolve from using the services offered by ICTs on the life cycle of other products or services. Today, the services offered by ICTs are usually realized by some kind of software. Therefore, software plays a significant role in the life cycles of many other products or services: software can be used to optimize product design, production processes, the end-of-life treatment, or the usage of other products or services. Unfortunately, these second-order effects are not as obvious as the first-order effects. Even harder to predict or analyze are third-order effects (or: systemic effects of ICT), because of the manifold systemic interdependencies, which require experienced knowledge from examiners. One example are rebound effects that may occur, if a specific optimization frees used resources, which can be used to produce more products, which then causes additional demand for these resources. This may in turn overcompensate the initially achieved savings. First-order effects deal with the term "Green IT", second- and third-order effects are connected with "Green by IT".

As can be seen from the previous paragraph, saving energy or resources by optimizing ICTs is only one aspect. The equally important aspect covers energy and resources that can be conserved by the usage of ICTs on other products and services. Seen from a broader point of view, not limited to resources or energy, the question is how negative impacts on ecology, society, human beings, and economy can be mitigated and how positive impacts on these can be promoted.

The problem here is that there is software that directly promotes sustainability aspects, like resource or energy efficiency, because it is its intended purpose, e.g. software that enables smart heating, smart lighting, smart logistics, paper free offices, etc. In these cases, it is relatively easy to assess second-order effects of the software. However, there is also multipurpose software, like word processors, spread sheets, or graphics software. For these, it is nearly impossible to assess second- or third-order impacts that result from using the software product, because software manufacturers usually do not know for which purposes their software product is used. Hence, a sustainable software product itself should have a low impact on SD and if it is its purpose, it should promote SD. These basic requirements for green or sustainable software are expressed in Definition 1.

Definition 1. "[*Green and Sustainable Software*] is software, whose direct and indirect negative impacts on economy, society, human beings, and environment that result from development, deployment, and usage of the software are minimal and/or which has a positive effect on sustainable development" [41].

However, a green and sustainable software product can only be achieved, if a developing organization is aware of negative and positive impacts on SD that will likely be caused when using it. In order to enable the various stakeholders to recognize these impacts, it is necessary to institutionalize their assessment and recognition in the applied software development processes. This makes sustainability issues manageable and puts software architects, designers, and developers in a position to optimize their software product accordingly. Additionally, it is necessary that the development process itself is environment-friendly. These two aspects are expressed in Definition 2.

Definition 2. *Green and Sustainable Software Engineering* is the art of developing green and sustainable software with a green and sustainable software engineering process. Therefore, it is the art of defining and developing software products in a way, so that the negative and positive impacts on sustainable development that result and/or are expected to result from the software product over its whole life cycle are continuously assessed, documented, and used for a further optimization of the software product [29].

4. The GREENSOFT Model

4.1. Overview of the model

The GREENSOFT Model is a conceptual reference model for "Green and Sustainable Software", which has the objective to support software developers, administrators, and software users in creating, maintaining, and using software in a more sustainable way. The model (see Fig. 1) comprises a holistic life cycle model for software products, sustainability criteria and metrics for software products, procedure models for different stakeholders, and recommendations for action, as well as tools that support stakeholders in developing, purchasing, supplying, and using software in a green and sustainable manner.

The reference model contains a *Life Cycle of Software Products*. That is, in contrast to traditional life cycles of software, geared to Life Cycle Thinking (abbr. LCT), which follows the motto: "from cradle to grave". LCT has the objective to assess the ecological, social, human, and economic compatibility of a product during its whole life cycle. It begins with the early stages of product development and ends with the product's disposal and recycling. The findings gained from these assessments can then be used for a balanced optimization of the product or for comparing a product with its competitors [42].

The second part of the GREENSOFT Model is called *Sustainability Criteria and Metrics*. It covers common metrics and criteria for the measurement of software quality [43] and it allows a classification of criteria and metrics for evaluating a software product's sustainability. Appropriate criteria and metrics may comprise models for the measurement of software quality, procedure models for software development, as well as methods borrowed from LCA [11,12]. Here, we distinguish direct criteria and metrics (related to first-order effects) from those which indirectly concern sustainability (related to second- and third-order effects).

The model component *Procedure Models* makes it possible to classify procedure models that cover acquisition and development of software, maintenance of IT systems, and user support. As an example, we proposed a generic extension for ambiguous software development processes that enables the systematic consideration of sustainability aspects during software development [29].

The last component of the model contains *Recommendations and Tools*. These support stakeholders with different professional skill levels in applying green or sustainable techniques in general, when developing, purchasing, administrating, or using software products. Possible roles are software developers, acquirers of software, administrators, as well as professional and private users [38,44].

4.2. The Life Cycle of Software Products

The *Life Cycle of Software Products* (see Fig. 2), included in the GREENSOFT model, is a LCT inspired product life cycle that can also be attributed with "from cradle-to-grave". Its objective is to enable stakeholders to assess impacts on SD according to the three different levels of impacts, as discussed in *What is Green and Sustainable Software?* section.

When working with these levels, stakeholders should be aware of holistically considering all levels, because: "Naturally it is easier to focus on first order impacts as they are immediate and obvious.

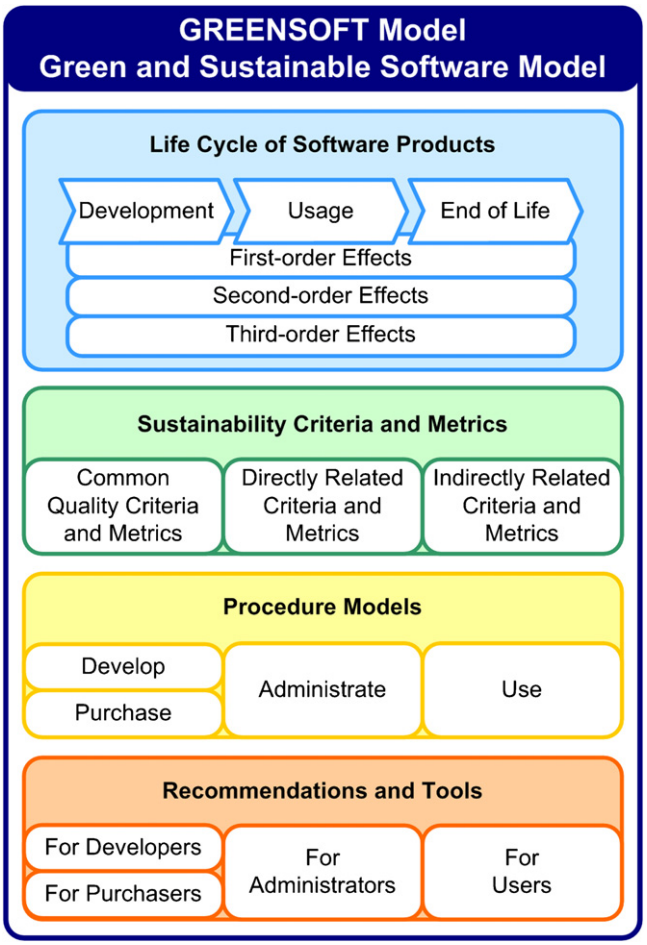

Fig. 1. The GREENSOFT Model, a reference model for "Green and Sustainable Software".

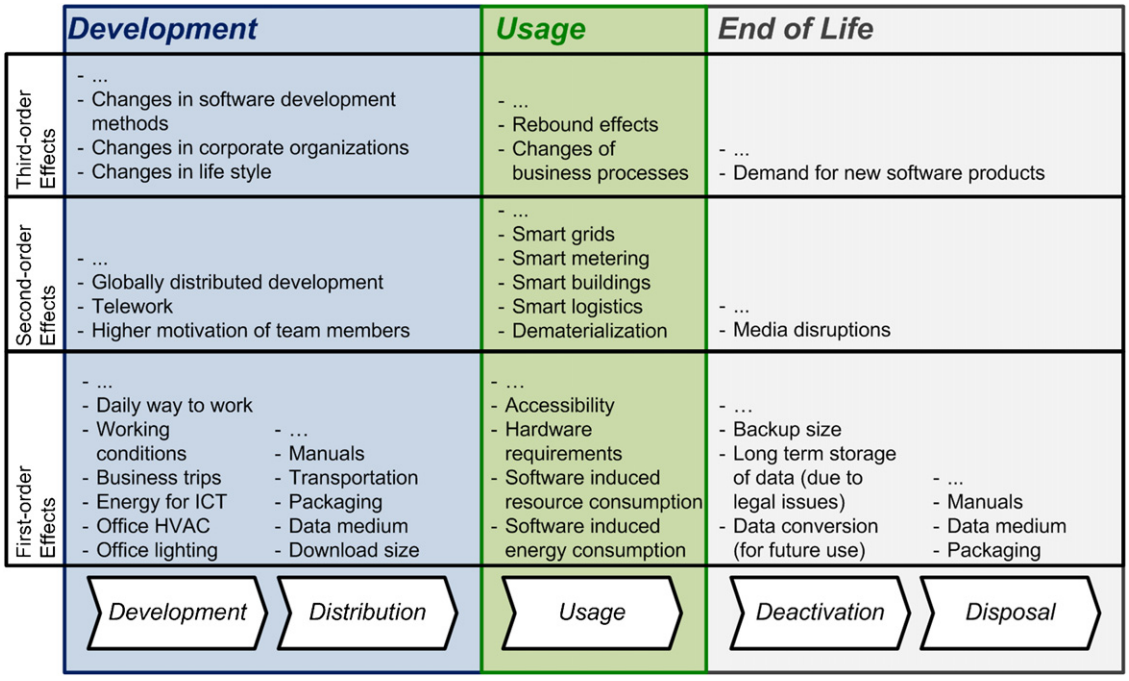

Fig. 2. Cradle-to-grave inspired product life cycle for software products, attributed to exemplary sustainability relevant effects of ICTs.

However the third order impacts might be the most threatening ones, and the most difficult ones to approach" [25]. Though, second- and third-order impacts also have huge potentials to promote SD, these still have to be recovered [3,23,45].

4.2.1. The development phase

The *Development Phase* accounts for impacts on SD that directly result from activities involved in software development, as well as indirectly involved activities, e.g. pro rata impacts of common corporate departments. Environmental impacts to be considered include, for example, electrical energy that is necessary to power the workstations of software developers and other employees, electrical energy and natural resources that are necessary to operate the IT infrastructure (e.g. networking devices, servers, and storages), energy that is necessary for heating and air conditioning, electrical energy that is necessary for offices lighting, or energy for transportation purposes like long distance business trips for meetings with customers and the development team and even the employees' daily way to work. Social impacts can be working conditions and payment of offshore workers (e.g. developers, type setters), which have consequences for the workers and their families. Some of these impacts can be mitigated by introducing teleworking and teleconferencing, or by replacing material products with adequate immaterial substitutes (second-order effects). This in turn may induce, e.g. changes in organizations, software development methods, or life styles (third-order effects). The development phase also accounts for impacts from software maintenance in the sense of bug solving, because this is also a software development activity and therefore should belong to the development phase.

4.2.2. The distribution phase and the disposal phase

The *Distribution Phase* accounts for impacts on SD that result from distributing the software product. This includes environmental impacts, e.g. of printed manuals (type of paper and ink), chosen means of transport, type and design of the retail and transport packaging (e.g. plastic, polyurethane foam, biodegradable material), or data medium (e.g. CD/DVD, USB memory stick). Furthermore, if the software product is offered as a download, which is common today, then its download size should be considered, as well as the electrical energy and material resources that are necessary to operate the required IT infrastructure.

The *Disposal Phase* accounts for impacts on SD that result from disposal and recycling of the afore mentioned material sub products.

4.2.3. The usage phase

The *Usage Phase* considers impacts that result from deploying, using, and maintaining the software product.

Here, maintaining means that administrators are in charge of installed software and support users in their organization. Thus, maintaining includes, e.g. the installation of software patches or updates, the configuration of software and computer systems, and the training of employees in regards to proper software usage. As an effect, properly trained users might need less time to complete tasks (which results in less energy consumption), configure the software system in a way that it consumes less power, or just switch their computer to suspend mode when they leave their workplace.

Beside these effects, software usage has several first-order effects regarding environmental sustainability.

In order to deliver its offered services, a computer program requires processing time, which in turn consumes electric energy. This may also require the consumption of services offered by other servers (consider, e.g. Data Base Management Systems, Enterprise Resource Planning systems, or simply the WWW service), which causes additional power consumption.

In addition, the update strategy of a software product (e.g. size and frequency of updates) influences data transfer, processing, and IT infrastructure, which are necessary to deliver updates. Combined, these cause further power and resource consumption.

State-of-the-art software systems usually require up-to-date and more powerful hardware than older software systems or previous versions. As a result, this causes hardware replacements in organizations as well as at home, when a new software product is introduced. On the one hand, new hardware is typically more power efficient than older hardware, but on the other hand it has to be taken into account that the production of the new hardware and the disposal of the old hardware causes vast amounts of resource and energy consumption [46]. Mining the necessary ores, e.g. in developing countries, where social and environmental standards are very low, leads to considerable social and environmental impacts, which sometimes even culminate in armed conflicts [15]. There are also reports about old and even non-functional hardware that is exported from industrial countries to developing countries, where it is reused but more often recycled under doubtful circumstances in so called backyard facilities or just deposited on waste disposal sites, causing damage to the environment and people's health [46].

The second- and third-order effects on SD that result from the usage phase, depend on the purpose of the software product and were briefly discussed in *What is Green and Sustainable Software?* section (smart-technologies, dematerialization).

4.2.4. The deactivation phase

If a software product is taken out of service, it is mostly necessary to convert the available data to a format that can be processed by the succeeding software product, or to make it accessible in some other ways. If the data cannot be converted easily, e.g. because it is stored in a proprietary data format, this may have an impact on economic sustainability of an organization. In this phase, even the backup size of data matters, e.g. if legal regulations require long-term storage of data.

4.3. Sustainability criteria and metrics

Our model has the ability to represent three categories of sustainability criteria and metrics for software products: *Common Quality Criteria and Metrics*, *Directly Related Criteria and Metrics*, and *Indirectly Related Criteria and Metrics*. The first relates to common quality criteria for software, which are well known from, e.g. [43]. The second comprises criteria and metrics that relate to first-order effects (effects of ICT supply). The last includes criteria and metrics geared towards second-order (effects of ICT use) and third-order effects (systemic effects of ICT). All criteria should also be classified according to the phases of our proposed software product life cycle. Additionally, it is also necessary to classify criteria and metrics according to the type of software.

Albertao et al. [26,27] interpret common software quality properties and associated metrics on the background of SD. They classify the quality properties into development, usage, and process related properties. However, they do not classify the effects according to the three tier model of effects of ICTs on SD. Notwithstanding this, their findings can be subsumed by the GREENSOFT Model.

The questions in the life cycle of the GREENSOFT Model are not: in which phases are metrics applied, or in which phases are measures taken, in order to improve the corresponding quality properties? Rather, the question is: in which life cycle phase can the related effects be observed?

Hence, the quality properties "Modifiability" and "Reusability" take effect in the *Development Phase*, whereas the properties "Portability", "Supportability", "Performance", "Dependability",

"Usability", and "Accessibility" take effect in the *Usage Phase*. The process related properties "Predictability" ("The team's ability to accurately estimate effort and cost upfront" [26]) and "Efficiency" ("The overhead of production processes over the bottom line value perceived by the customer" [26]) also take effect in the *Development Phase*, as well as the "Project's Footprint" ("Natural resources and environmental impact used during software development" [26]).

The property "Portability" is interpreted by Albertao et al. on the background of hardware obsolescence, which means that the lifetime of hardware should be prolonged to the end of its useful lifetime, instead of causing its early replacement due to hardware requirements imposed by a software product. According to Hilty [46] based on a LCA study by Eugster et al. [47], desktop PCs should not be used for less than approx. 5 years. After 5 years of use, the ecological impacts resulting mainly from the energy demand of the usage phase outweigh those of the production phase. These findings are also supported by a recent LCA study by a manufacturer of ICTs [48]. This is different for servers, because a server in 24/7 operation mode reaches the point of equilibrium earlier [46], according to the figures presented by [49] after approx. 1 year of use. Due to the fact that there will be higher rates of renewable energy and more energy efficient hardware in the future, it will take even longer until the environmental impacts of the usage phase outperform these of the production phase. For that reason, software induced hardware obsolescence is of particular importance. Hence, hardware obsolescence should be a genuine quality property of green and sustainable software, which belongs to the *Directly Related Criteria and Metrics* model part.

Another directly related quality property is energy efficiency. This is not the same as run time efficiency or performance, because its goal is to optimize energy consumption in relation to delivered service items. For a software running on a single computer, energy efficiency and performance may be closely related, but there may be greater differences for distributed systems. In this area, consider, e.g. service level agreements or performance requirements that necessitate additional servers to handle peak loads. Here, it may be possible to increase energy efficiency by relaxing performance requirements for peak loads. Another possibility may be the relaxation of required service quality down to a level that is still acceptable for users. Depending on the area of application, it may be possible to use approximations instead of accurate calculations, which may require less processing. Consider, e.g. a search engine that delivers a huge amount of accurate search results compared to one that ensures this accuracy only for the topmost results that are used most widely by the users but is less accurate regarding the following results. Another example are mathematical calculations, which may be only as accurate as they are needed to be acceptable for the purpose of the software application.

In addition to these software artifacts and development process related properties, there are also properties, which belong to the developing organization. Organizations that develop green and sustainable software should commit themselves to environmental and social responsibility, expressed, e.g. in environmental and social responsibility statements, their commitment to international labor standards [50], or the application of environmental management systems [51]. These commitments should also cover environmental and social standards throughout the entire supply chain of all products and services, which are necessary to produce, advertise, distribute, and dispose/recycle the software product or parts of it.

Indirectly Related Criteria and Metrics for green and sustainable software address second- and third-order effects induced by a software product. Using a software product may achieve energy and resource savings in other branches or usage areas. Although it is hard to identify these second- and third-order effects, there may be effects that can be measured and approximated anyway, i.e. when the software product has a special purpose like smart heating and air-conditioning of buildings. Even harder to recognize are social effects connected with a software product. Because these effects specifically depend on the software product's purpose and area of application, it is not possible to define appropriate metrics or criteria to recognize these in general. Examples are how social networks change the way to communicate and to interact, and the way production chains are altered, since ICTs almost allow world-wide production in real time.

4.4. Procedure models

Starting from our definitions, *Green and Sustainable Software Engineering* produces *Green and Sustainable Software* in an environmental-friendly and sustainable way. Consequently, during the engineering process, the whole life cycle of the engineered software product has to be taken into account, as well as the circumstances under which it will be produced.

4.4.1. Sub-procedure model "Develop"

In this section, we present an example of a software development process that fits into the category *Develop* of the procedure model part of the GREENSOFT Model.

This example [29] (see Fig. 3) does not implement a complete software development process. Instead, it proposes several enhancements for arbitrary software development processes that enable stakeholders to recognize impacts, which result from "producing" the software product, and impacts, which result from using the software product. The proposed enhancements are: *Sustainability Reviews & Previews*, *Process Assessment*, *Sustainability Journal*, and the so called *Sustainability Retrospective* [29]. In principle, the enhancements form a continuous improvement cycle that deals with sustainability issues. *Process Assessment* helps to optimize the sustainability of the "production" process, whereas *Sustainability Reviews & Previews* help to optimize the sustainability of the evolving software product. Both efforts are combined by the *Sustainability Retrospective*, so that finally impacts over the whole life cycle of the software product are covered.

Sustainability Reviews & Previews take a look at the work done, assess outcomes according to sustainability issues, and develop measures, which are realized until the next *Sustainability Review & Preview* in order to optimize the sustainability of the software product under development. These reviews take software aspects, like requirements, architecture, or coding into account that have impacts on sustainability. In *Sustainability Reviews & Previews*, which take the role of a formative evaluation, not only these software aspects are taken into account, but also impacts that result from the development process itself [29].

The *Process Assessment* activity continuously monitors the development process. Therefore, different data from the development process is gathered in order to assess its impacts on SD, and to identify factors that should be optimized in order to improve the process. This data can also be used as a basis to perform a LCA of the software product [29].

The *Sustainability Retrospective* sums up the data collected by *Sustainability Reviews & Previews* and *Process Assessment*, assesses the overall impact on SD of the software product, and looks for ways to improve the sustainability of upcoming software projects and their software products. Further outcomes of the *Sustainability Retrospective* are, e.g. assessments and group reflections of impacts on SD of the developed software product and the development process, decisions for future projects, lessons learned, or best practices regarding sustainability issues of software products and development processes [29].

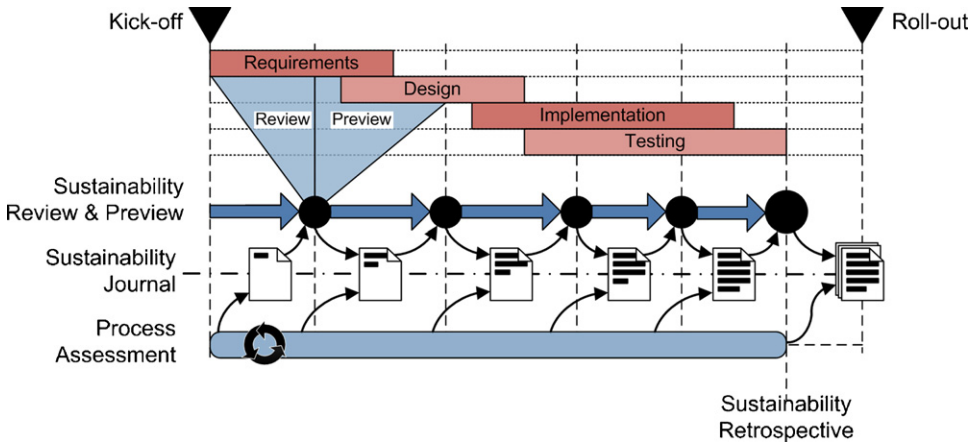

Fig. 3. Example for enhancing software development processes that fits into the procedure model part "Develop" [29].

It should be mentioned, that this procedure model sets only an organizational framework that helps managing sustainability of a software product. Thus, there is no guarantee that the resulting software products are more sustainable than they would have been without applying this process.

Besides reflecting the proposed life cycle of a software product, there are further methods that support software architects, designers, and developers in producing green and sustainable software applications. On the one hand, there are tools that automatically calculate software metrics from source code or compiled artifacts. As was shown above, these metrics and their corresponding quality properties can also be interpreted regarding sustainability. On the other hand, producing ecologically sound, resource and energy efficient software is also an issue nowadays. For this purpose early processing time estimations, energy consumption estimations, and energy consumption measurements may be appropriate.

Early processing time estimations can be obtained from a software performance estimation method, introduced by Smith and Williams [52]. This method estimates performance values already in early design stages of a software product. It starts with rough estimations based on early UML sequence diagrams [53] and refines these estimations as the UML model evolves. This results in a so called software execution model. The system execution model models the target hardware platform with its limited resources (e.g. CPU cores, net IO, disk IO). The software execution model can be applied to the system execution model with a discrete event simulator in order to get performance estimations for realistic workloads [52]. This approach helps to design software architectures that perform well on a specific hardware platform, and it helps to identify design flaws that reduce runtime efficiency. Currently, there is no software tool available that allows the estimation of energy consumption in these early design stages. So we recommend using the software performance approach as an indicator of energy efficiency.

In a software project, when the first deployable software artifacts are available, it is easily possible to measure their energy consumption, either by measuring the energy consumption directly [39] or by using performance monitor counters of modern processors as indicators [40]. These measurements complement the early estimations and can provide further indications on software components that induce high energy consumption and should therefore be optimized with priority.

Based on these software architecture and software development centric measures, software developing organizations should also estimate the total energy and resource demand that is expected according to their projected number of installations or sales figures of a specific product and associated estimated usage scenarios. These estimations should not only include first-order impacts of the usage phase, but also first-order impacts of the other phases, especially of the distribution phase. As a further step and depending on the type of software, it may also be possible to estimate second- and third-order impacts respectively. These can be used to substantiate the necessity to improve the sustainability of a software product from a broader point of view.

4.4.2. Sub-procedure model "Purchase"

Our outline of an exemplary procedure model that fits into the *Purchase* category focuses mainly on governmental organizations and large enterprises that use structured tendering procedures in large scale procurement projects. Especially the large market power of governmental organizations should not be underestimated and can be used to pursue sustainability goals according to national (e.g. [54]) and international agreements (e.g. [55,56]).

A typical procurement process has the following steps: define subject matter, define requirements, select bidders, evaluate bids, and conclude contract [51,57]. Due to legal reasons, it is necessary that sustainability issues, i.e. ecological and social requirements, are clearly stated in the tender's subject matter, in specifications and in contract performance clauses [57]. Furthermore, the entire product life cycle of the software product should be addressed, as well as the entire supply chain. Our proposal for a sustainable software procurement process can be divided into two fields: the procurement of custom software products and the procurement of standard software products.

In both scenarios, bidders should, in principle, be able to deliver the requested product in the required quality. Hence, bidders may be preselected with appropriate criteria, like the company's social and environmental responsibility (e.g. expressed in environmental and social responsibility statements), their commitment to international labor standards [50] or the application of environmental management systems [51].

For custom software products, non-functional requirements like energy efficiency or requirements addressing mitigation of IT infrastructure obsolescence, which may be induced by the tendered software product, can be defined. However, it is necessary to provide applicable measurement methods and acceptable maximum measurement values for bidders. From a practical point of view, this is only possible if a comparable software product is available that can be used as reference, e.g. if a legacy software system is replaced with a new one. If such values cannot be provided, purchasers

can define contract performance clauses that pledge the contractor to establish measures, which encourage a sustainable software product. This includes the application of a sustainable software development process, environment management systems, or the assertion of social standards along the entire supply chain (includes also subcontractors).

For standard software products, energy efficiency and hardware obsolescence criteria can be either used as technical requirements or as award criteria. Usually, their limit values and weighting in the selection procedure must be documented in the tendering documents, which means that these cannot be altered after the tendering procedure has started.

Purchasing software or purchasing the appropriate hardware for a software product is also of high relevance for home users. It is clear, that home users, as well as purchasers of micro and small enterprises, require information on sustainability issues that can be obtained easily. This may be accomplished by printing the according information on product boxes or product sheets. In this way, the customers can make informed decisions on which product fits their needs best. Hence, there may be a label for software products similar to the ENERGY STAR®[5] that indicates whether a software product is energy efficient or meets certain sustainability requirements in the future.

4.4.3. Sub-procedure model "Administrate"

In our model, "administrate" means making software available by installing, configuring, and maintaining it. This also covers educating and training users, who work with the software in an organization.

Configuring software products covers only configuration from an organizational point of view, here called "macro-configuration". Configurations that can be done by users without the need for special system permissions, e.g. installing add-ons for browsers, configuring a word processor via its preferences dialog, etc., are covered as "micro-configuration" by the sub-procedure model *Use*. Hence, the macro-configuration covers system configuration and maintenance of desktop PCs, thin-clients, as well as servers and virtualized data centers. Especially, the various publications on data center energy efficiency fit into this model part (e.g. [58–60]).

A minimalistic procedure model should implement a continuous improvement cycle, which follows the plan-do-check-act paradigm. Here, i.e. energy efficiency, energy consumption, and resource consumption should be checked regularly in order to improve these with appropriate measures. This is not merely limited to data center operations, but rather includes networking infrastructure, desktop computers, installed software, and users, served by an IT service division of an organization. Especially in organizations that implement service desks in line with the IT Infrastructure Library [61] in order to support their IT users, the mission of service desks also is to advise users proactively on software configuration and usage regarding sustainability issues like energy and resource conservation.

4.4.4. Sub-procedure model "Use"

Procedures for users, either professional or home users, related to green and sustainable software, cannot be put into a static workflow or procedure model, because these are mainly ad-hoc procedures. Nevertheless, similarly to the sub-procedure model component *Administrate*, a continuous improvement cycle can be applied intuitively. In this context, users should reflect on each of their actions that may have negative impacts on SD in order to search for appropriate guidance that helps mitigating these negative impacts.

As already mentioned in *Sub-procedure model "Administrate"* section, users do some kind of "micro-configuration" like installing add-ons in web browsers, configuring the word processor, etc. and they use software products. However, because of the absence of a more formalized process and the ad-hoc character of the actions, procedures are more or less represented directly as recommendations for action like guidelines or checklists.

4.5. Recommendations and tools

Recommendations and Tools address stakeholders with different roles. General roles considered distinctly by the GREENSOFT Model are: *Developer*, *Purchaser*, *Administrator*, and *User*. However, there may be more specialized roles, like Requirements Engineer, Software Architect, Web Administrator, or Application Deployer. In principle, these can be subsumed by the general roles mentioned before.

Recommendations and Tools support stakeholders with different skill levels in applying green and sustainable techniques when developing, administrating or using software products. Recommendations can be guidelines, checklists, best practice examples, implementation reports, etc. Tools can be software tools, but also any other tool, like paper-based data collection sheets. There are plenty of recommendations available, e.g. on the Internet, but unfortunately these are hard to find. Hence, a specialized Internet search engine or knowledge base would make it easier to find them [44].

Example recommendations for developers deal with website optimization [38,62], provide guidance on how to design and develop energy efficient software [39,40,63,64], or give general hints on how to design and develop resource efficient software [65]. Available tools focus on energy or resource efficiency of software or services [39,66]. For administrators, there are guidelines available that describe best practices for energy efficient data centers [59,60,67,68]. Some of these guidelines are accompanied by appropriate tools [69]. For purchasers, there are guidelines available that describe sustainable procurement of IT equipment, in accordance with tendering regulations [51,57]. For users, there are guidelines and tools available that support users in making informed decisions on hardware, software, and services they use or plan to use [32,33,70,71]. Other tools like grano.la[6] directly help saving energy by reconfiguring the power management of the computer system according to current performance demands.

5. Discussion and conclusion

The main objective of our proposed GREENSOFT Model is to structure concepts, strategies, activities, and processes of Green IT and hereby especially of green and sustainable software and its engineering. Our model acts as a reference model, which helps to organize and classify research results, actions, frameworks, process models, etc. Additionally, the GREENSOFT Model involves an inherent roadmap and depicts which direction future developments regarding the interconnection of ICT and SD can take. The model also suggests how "conventional" models of ICT, e.g. standard software process models or tendering procedures, can be enriched regarding sustainability.

Consequently, at first we defined what *Green and Sustainable Software* is and what *Green and Sustainable Software Engineering*

[5] ENERGY STAR® is a registered trademark owned by United States Environmental Protection Agency (EPA); http://www.energystar.gov/; http://www.eu-energystar.org/ (2011-03-15).

[6] http://www.grano.la/ (2011-03-23).

means. Starting with these definitions we described in our contribution the four main parts of the GREENSOFT Model: (1) the life cycle of a software product regarding a cradle-to-grave approach and (2) criteria and metrics for direct and indirect effects of software on SD. Furthermore, we (3) included procedure models for developing, purchasing, operating and using sustainably sound software in a sustainably sound way and (4), in order to put the model into practice, the GREENSOFT Model contains a framework for recommendations for action and tools.

We showed that a software product is not merely made up of software artifacts, but rather of many other products and services that are involved in a software products' life cycle. All these products and services have plenty of impacts on SD that must be considered in order to figure out if a software product is green or sustainable. Furthermore, using the software product leads to effects on SD. Here, especially second- and third-order effects can either exceed or even out-Herod first-order effects.

Our GREENSOFT Model takes those effects, the life cycle of software, different user roles, and different activities into account. Also aspects like "Green IT" vs. "Green by IT" are integrated. Different approaches regarding Green IT, Sustainable IT, ICT for Sustainability, etc. can be structured and classified into the model. Furthermore, the model can be adopted in order to unfold software related Green IT activities in business and science.

However, in every decision regarding Green IT, one must bear in mind that every additional software product consumes additional energy. Consequently, these effects must be compared with the benefits using this special software product [45] resp. [46]. Our GREENSOFT Model helps to find a comprehensive solution.

Acknowledgements

This paper evolved from the research and development project Green Software Engineering (GREENSOFT), which is sponsored by the German Federal Ministry of Education and Research under reference 17N1209. The contents of this document are the sole responsibility of the authors and can under no circumstances be regarded as reflecting the position of the German Federal Ministry of Education and Research.

References

[1] United Nations General Assembly, Report of the World Commission on Environment and Development, Our Common Future, UN Document No. A/43/427 English, New York, 1987, http://daccess-ods.un.org/TMP/4738853.html (accessed 2011-03-09).
[2] K. Fichter, Sustainable business strategies in the Internet economy , in: L.M. Hilty (Ed.), Sustainability in the Information Society, Metropolis-Verlag, Marburg, 2001.
[3] L. Erdmann, L.M. Hilty, J. Goodman, P. Arnfalk, The Future Impact of ICTs on Environmental Sustainability, 2004, http://ftp.jrc.es/EURdoc/eur21384en.pdf (accessed 2010-04-26).
[4] U.S. EPA, Report to Congress on Server and Data Center Energy Efficiency Public Law 109–431, 2007, http://hightech.lbl.gov/documents/DATA_CENTERS/EPA-datacenters.pdf (accessed 2010-10-15).
[5] J.G. Koomey, Estimating total Power Consumption by Servers in the U.S. and the World, Final Report, February 15, 2007, 2007, http://www.greenbiz.com/sites/default/files/document/CustomO16C45F77412.pdf (accessed 2010-04-26).
[6] W. Göhring, The memorandum "Sustainable Information Society", in: P. Minier, A. Susini (Eds.), Sh@ring, Proceedings of the 18th International Conference "Informatics for Environmental Protection", EnviroInfo 2004, 21–23 October, 2004, Geneva, CERN, Geneva (Switzerland), Éditions du Tricorne, Genève, 2004, pp. 278–286.
[7] V. Coroama, L.M. Hilty, Energy consumed vs. energy saved by ICT – a closer look, in: V. Wohlgemuth, B. Page, K. Voigt (Eds.), Environmental Informatics and Industrial Environmental Protection: Concepts, Methods and Tools, EnviroInfo 2009; 23rd International Conference on Informatics for Environmental Protection; Proceedings of the 23rd International Conference Environmental Informatics – Informatics for Environmental Protection, Sustainable Development and Risk Management, September 09-11, 2009, HTW Berlin, University of Applied Sciences, Germany, vol. 2: Workshops, vol. 2, Shaker, Aachen, 2009, pp. 353–361.
[8] S. Naumann, Sustainability informatics – a new subfield of applied informatics? in: A. Möller, B. Page, M. Schreiber (Eds.), Environmental Informatics and Industrial Ecology; 22nd International Conference on Informatics for Environmental Protection; EnviroInfo 2008; Proceedings of the 22nd International Conference Environmental Informatics – Informatics for Environmental Protection, Sustainable Development and Risk Management, September 10–12, 2008, Leuphana University Lueneburg, Germany, Shaker, Aachen, 2008, pp. 384–389.
[9] F. Berkhout, J. Hertin, Impacts of Information and Communication Technologies on Environmental Sustainability: Speculations and Evidence, Report to the OECD, 2001, http://www.oecd.org/dataoecd/4/6/1897156.pdf (accessed 2011-03-02).
[10] L.M. Hilty, CO_2 reduction with ICT: prospects and barriers, in: O. Hryniewicz, J. Studziński, A. Szediw (Eds.), Environmental Informatics and Systems Research, EnviroInfo Warsaw 2007; The 21st International Conference on "Informatics for Environmental Protection", Warsaw, Poland, September 12–14, 2007, vol. 1, Shaker, Aachen, 2007, pp. 35–42.
[11] Deutsches Institut für Normung e.V., Environmental Management – Life cycle Assessment – Principles and Framework (ISO 14040:2006); German and English version EN ISO 14040:2006, Beuth, Berlin, vol. 13.020.10, 2009.
[12] Deutsches Institut für Normung e.V., Environmental Management – Life Cycle Assessment – Requirements and Guideline (ISO 14044:2006); German and English version EN ISO 14044:2006, Beuth, Berlin, ICS 13.020.10, 2006.
[13] C. Fuchs, The implications of new information and communication technologies for sustainability , Environment Development and Sustainability 10 (3) (2008) 291–309.
[14] L.M. Hilty, A. Köhler, F. Von Schéele, R. Zah, T. Ruddy, Rebound effects of progress in information technology , Poiesis & Praxis: International Journal of Technology Assessment and Ethics of Science 4 (1) (2006) 19–38.
[15] S. Behrendt, W. Kahlenborn, M. Feil, C. Dereje, B. Raimund, D. Ruth, S. Michael, Rare Metals, Measures and Concepts for the Solution of the Problem of Conflict-aggravating Raw Material Extraction – The Example of Coltan, Umweltbundesamt, 2007.
[16] S. Bormann, L. Plank, Under pressure , in: Working Conditions and Economic Development in ICT Production in Central and Eastern Europe, WEED, Berlin, 2010.
[17] J. Chan, C. Ho, The dark side of cyberspace , in: Inside the Sweatshops of China's Computer Hardware Production, WEED, Berlin, 2008.
[18] S. Prakash, A. Manhart, Socio-economic assessment and feasibility study on sustainable e-waste management in Ghana, Commissioned by the Inspectorate of the Ministry of Housing, Spatial Planning and the Environment of the Netherlands (VROM-Inspectorate) and the Dutch Association for the Disposal of Metal and Electrical Products (NVMP), Freiburg, 2010, http://www.oeko.de/oekodoc/1057/2010-105-en.pdf (accessed 2011-03-03).
[19] A. Magashi, M. Schluep, e-Waste Assessment Tanzania, UNIDO e-waste initiative for Tanzania, Final Report, 2011, http://ewasteguide.info/files/Magashi_2011_CPCT-Empa.pdf (accessed 2011-03-04).
[20] K. Brigden, I. Labunska, D. Santillo, P. Johnston, Chemical contamination at e-waste recycling and disposal sites in Accra and Korforidua,Ghana, Amsterdam, 2008, http://www.greenpeace.org/raw/content/international/press/reports/chemical-contamination-at-e-wa.pdf (accessed 2011-03-04).
[21] R. Widmer, H. Oswald-Krapf, D. Sinha-Khetriwal, M. Schnellmann, H. Böni, Global perspectives on e-waste , Environmental and Social Impacts of Electronic Waste Recycling, Environmental Impact Assessment Review 25 (5) (2005) 436–458.
[22] R. Kahhat, E. Williams, Product or waste? Importation and end-of-life processing of computers in Peru , Environmental Science & Technology 43 (15) (2009) 6010–6016.
[23] The Climate Group, GeSI, Global e-Sustainability Initiative, SMART 2020: Enabling the Low Carbon Economy in the Information Age, 2008, http://www.smart2020.org/_assets/files/02_Smart2020Report.pdf (accessed 2011-03-04).
[24] D. Mocigemba, Sustainable computing , Poiesis & Praxis: International Journal of Technology Assessment and Ethics of Science 4 (3) (2006) 163–184.
[25] S. Abenius, Green IT & Green Software – time and energy savings using existing tools, in: V. Wohlgemuth, B. Page, K. Voigt (Eds.), Environmental Informatics and Industrial Environmental Protection: Concepts, Methods and Tools, EnviroInfo 2009; 23rd International Conference on Informatics for Environmental Protection; Proceedings of the 23rd International Conference Environmental Informatics – Informatics for Environmental Protection, Sustainable Development and Risk Management, September 09–11, 2009, HTW Berlin, University of Applied Sciences, Germany, Vol. 1: Sessions für Band 1, Shaker, Aachen, 2009, pp. 57–66.
[26] F. Albertao, J. Xiao, C. Tian, Y. Lu, K.Q. Zhang, C. Liu, Measuring the sustainability performance of software projects, in: IEEE Computer Society (Ed.), 2010 IEEE 7th International Conference on e-Business Engineering (ICEBE 2010), Shanghai, China, 2010, pp. 369–373.
[27] F. Albertao, Sustainable Software Engineering, 2004, http://www.scribd.com/doc/5507536/Sustainable-Software-Engineering#about (accessed 2010-11-30).
[28] H.-K. Arndt, S. Lau, A. Strehl, Sustainability of information and communication systems (ICS), in: V. Wohlgemuth, B. Page, K. Voigt (Eds.), Environmental Informatics and Industrial Environmental Protection: Concepts, Methods and Tools, EnviroInfo 2009; 23rd International Conference on Informatics for

Environmental Protection; Proceedings of the 23rd International Conference Environmental Informatics – Informatics for Environmental Protection, Sustainable Development and Risk Management, September 09–11, 2009, HTW Berlin, University of Applied Sciences, Germany, Vol. 1: Sessions für Band 1, Shaker, Aachen, 2009, pp. 67–74.
[29] M. Dick, S. Naumann, Enhancing software engineering processes towards sustainable software product design, in: K. Greve, A.B. Cremers (Eds.), EnviroInfo 2010, Integration of Environmental Information in Europe, Proceedings of the 24th International Conference on Informatics for Environmental Protection, October 6–8, 2010, Cologne/Bonn, Germany, Shaker, Aachen, 2010, pp. 706–715.
[30] A. Kansal, F. Zhao, J. Liu, N. Kothari, A.A. Bhattacharya, Virtual machine power metering and provisioning, in: Proceedings of the 1st ACM Symposium on Cloud Computing, ACM, Indianapolis, Indiana, USA, 2010, pp. 39–50.
[31] J.L. Zapico, N. Brandt, M. Turpeinen, Environmental Metrics, The main opportunity from ICT for industrial ecology , Journal of Industrial Ecology 14 (5) (2010) 703–706.
[32] N. Amsel, B. Tomlinson, Green Tracker: a tool for estimating the energy consumption of software, in: Association for Computing Machinery (Ed.), CHI EA 10: Proceedings of the 28th of the International Conference Extended Abstracts on Human Factors in Computing Systems, ACM, New York, 2010, pp. 3337–3342.
[33] S. Naumann, S. Gresk, K. Schäfer, How green is the web? Visualizing the power quality of websites, in: A. Möller, B. Page, M. Schreiber (Eds.), Environmental Informatics and Industrial Ecology, 22nd International Conference on Informatics for Environmental Protection; EnviroInfo 2008; Proceedings of the 22nd International Conference Environmental Informatics – Informatics for Environmental Protection, Sustainable Development and Risk Management, September 10–12, 2008, Leuphana University Lueneburg, Germany, Shaker, Aachen, 2008, pp. 62–65.
[34] E. Capra, G. Formenti, C. Francalanci, S. Gallazzi, The impact of MIS software on IT energy consumption, in: 18th European Conference on Information Systems, June 7–9, 2010, Pretoria, South Africa, 2010, http://web.up.ac.za/ecis/ECIS2010PR/ECIS2010/Content/Papers/0073.R1.pdf (accessed 2010-10-25).
[35] B. Steigerwald, R. Chabukswar, K. Karthik, D.V. Jun, Creating Energy-Efficient Software, 2007, http://software.intel.com/en-us/articles/creating-energy-efficient-software-part-1/ (accessed 2011-01-19).
[36] M. Nottingham, Caching Tutorial, for Web Authors and Webmasters, 2010, http://www.mnot.net/cache_docs/ (accessed 2010-04-19).
[37] A.B. King, Website optimization , in: Speed, Search Engine & Conversion Rate Secrets, 1st ed., O'Reilly, Beijing, 2008.
[38] M. Dick, S. Naumann, A. Held, Green web engineering, a set of principles to support the development and operation of "Green" websites and their utilization during a website's life cycle, in: J. Filipe, J. Cordeiro (Eds.), WEBIST 2010 – Proceedings of the Sixth International Conference on Web Information Systems and Technologies, vol. 1, Valencia, Spain, April 07–10, 2010, INSTICC Press, Setúbal, 2010, pp. 48–55.
[39] Intel Corporation, Intel® Energy Checker SDK, http://software.intel.com/en-us/articles/intel-energy-checker-sdk/ (accessed 2011-03-04).
[40] S. Wang, H. Chen, W. Shi, SPAN: a software power analyzer for multicore computer systems , SUSCOM 1 (1) (2011) 23–34.
[41] M. Dick, S. Naumann, N. Kuhn, A model and selected instances of green and sustainable software, in: J. Berleur, M.D. Hercheui, L.M. Hilty (Eds.), What Kind of Information Society? Governance, Virtuality, Surveillance, Sustainability, Resilience, 9th IFIP TC 9 International Conference, HCC9 2010 and 1st IFIP TC 11 International Conference, CIP 2010, Held as Part of WCC 2010, Brisbane, Australia, September 20–23, 2010; Proceedings, IFIP International Federation for Information Processing, Berlin, Heidelberg, 2010, pp. 248–259.
[42] U. Tischner, B. Dietz, S. Maßelter, E. Schmincke, M. Prösler, F. Rubik, B. Hirschl, How to do EcoDesign? A Guide for Environmentally and Economically Sound Design, Verlag form, Frankfurt am Main, 2000.
[43] International Organization for Standardization, Software Engineering – Software Product Quality Requirements and Evaluation (SQuaRE) – Guide to SQuaRE, vol. 35.080, 2005.
[44] J. Fischer, S. Naumann, M. Dick, Enhancing sustainability of the software life cycle via a generic knowledge base, in: K. Greve, A.B. Cremers (Eds.), EnviroInfo 2010, Integration of Environmental Information in Europe; Proceedings of the 24th International Conference on Informatics for Environmental Protection, October 6–8, 2010, Cologne/Bonn, Germany, Shaker, Aachen, 2010, pp. 716–725.
[45] OECD Information Technology Outlook 2010, OECD Publishing, OECD, Paris, 2010.
[46] L.M. Hilty, Information technology and sustainability , in: Essays on the Relationship between ICT and Sustainable Development, Books on Demand, Norderstedt, 2008.
[47] M. Eugster, R. Hischier, H. Duan, Key Environmental Impacts of the Chinese EEE-Industry, A Life Cycle Assessment Study, Final Report, St. Gallen, 2007, http://library.eawag-empa.ch/empa_publications_2007_open_access/EMPASG20070177.pdf (accessed 2011-01-21).
[48] FUJITSU Technology Solutions GmbH, Life Cycle Assessment and Product Carbon Footprint, Fujitsu ESPRIMO E9900 Desktop PC, White Paper, 2010, http://fujitsu.fleishmaneurope.de/wp-content/uploads/2010/12/Whitepaper-LCA-PCF-ESPRIMO-E9900.pdf (accessed 2011-03-16).
[49] FUJITSU Technology Solutions GmbH, Life Cycle Assessment and Product Carbon Footprint, PRIMERGY TX 300 S5 and PRIMERGY RX 300 S5 Server, White Paper, 2010, http://fujitsu.fleishmaneurope.de/wp-content/uploads/2010/12/LCA_PCF-Whitepaper-PRIMERGY-TX-RX-300-S5.pdf (accessed 2011-03-30).
[50] ILO – International Labour Organization, International Labour Standards, ILO is a specialized agency of the United Nations, http://www.ilo.org/global/standards/lang–en/index.htm (accessed 2011-03-17).
[51] P. Tepper, M. Hidson, S. Clement, M. Anglada, Sustainable Procurement Guidelines for Office IT Equipment, Product Sheet, The Copy-and-paste Guide to Sustainable Tendering for Office IT Equipment, 2008, http://www.unep.fr/scp/sun/facility/reduce/procurement/PDFs/SP%20guidelines%20office%20IT%20-%20Product%20Sheet%20-%20final.pdf (accessed 2011-02-22).
[52] C.U. Smith, L.G. Williams, Performance solutions , in: A Practical Guide to Creating Responsive, Scalable Software, Addison-Wesley, Boston, 2002.
[53] OMG, Object Management Group, OMG Unified Modeling Language™ (OMG UML), Superstructure, 2010, http://www.omg.org/spec/UML/2.3/Superstructure/PDF/ (accessed 2011-03-11).
[54] BMWi – German Federal Ministry of Economics and Technology, Action plan "Germany: Green IT Pioneer", Berlin, 2008, http://www.bmwi.de/English/Redaktion/Pdf/action-plan-green-it-pioneer,property=pdf,bereich=bmwi,sprache=en,rwb=true.pdf (accessed 2011-03-15).
[55] United Nations, Kyoto Protocol to the United Nations Framework Convention on Climate Change, 1998, http://unfccc.int/resource/docs/convkp/kpeng.pdf (accessed 2011-03-15).
[56] European Commission, A Roadmap for Moving to a Competitive Low Carbon Economy in 2050, Communication from the Commission to the European Parliament, the Council, the European Economic and Social Committee, and the Committee of the Regions, Brussels, 2011, http://ec.europa.eu/clima/documentation/roadmap/docs/com_2011_112_en.pdf (accessed 2011-03-15).
[57] F. Butollo, J. Kusch, T. Laufer, Buy IT Fair, Guideline for Sustainable Procurement of Computers, Berlin, 2009, http://procureitfair.org/publications-en/Publication_3125/at_download/fullfile (accessed 2011-03-17).
[58] BITKOM Bundesverband Informationswirtschaft Telekommunikation und neue Medien e.V., Energy Efficiency in the Data Center, A Guide to the Planning, Modernization and Operation of Data Centers, Berlin, 2008, http://www.bitkom.org/files/documents/Energy_Efficiency_in_the_Data_Center_web.pdf (accessed 2010-11-17).
[59] European Commission, Best Practices for the EU Code of Conduct on Data Centres, 2008, http://re.jrc.ec.europa.eu/energyefficiency/pdf/CoC%20data%20centres%20nov2008/Best%20Practices%20v1.0.0%20-%20Release.pdf (accessed 2010-10-15).
[60] U.S. Department of Energy, Data Center Energy Assessment Process, Save Energy Now, http://www1.eere.energy.gov/industry/datacenters/pdfs/data_center_assessment_process.pdf (accessed 2011-03-22).
[61] Service Support, ITIL; [The Key to] Managing IT Services, 11th ed., TSO (The Stationery Office), London, 2005.
[62] T. Theurer, Performance Research, Part 2, Browser Cache Usage – Exposed! 2007, http://www.yuiblog.com/blog/2007/01/04/performance-research-part-2/ (accessed 2010-07-01).
[63] P. Larsson, Energy-Efficient Software Guidelines, White Paper, 2008, http://software.intel.com/en-us/articles/energy-efficient-software-guidelines/ (accessed 2011-01-19).
[64] P. Larsson, Energy-Efficient Software Checklist, 2009, http://software.intel.com/en-us/articles/energy-efficient-software-checklist/ (accessed 2011-01-19).
[65] H. Shojaee, Rules for Being a Green Software Engineer, Ship Software OnTime! The blog that Helps You Build Great Software, 2007, http://shipsoftwareontime.com/2007/12/24/rules-for-being-a-green-software-engineer/ (accessed 2010-05-03).
[66] Google Project Hosting, Page Speed, mod_pagespeed, Open Source Project, http://code.google.com/intl/de/speed/page-speed/docs/module.html (accessed 2011-03-23).
[67] Dimension 85, Best Practices V1.0 for the EU Code of Conduct for Data Centres Energy Efficiency, Summary, 2009, http://dimension85.com/EN/best-practices-summary.pdf (accessed 2010-10-15).
[68] BMU – German Federal Ministry for the Environment, Nature Conservation and Nuclear Safety, Energy-Efficient Data Centres, Best-Practice Examples from Europe, the USA and Asia, 2010, http://www.bmu.de/files/pdfs/allgemein/application/pdf/broschuere_rechenzentren_en_bf.pdf (accessed 2010-10-15).
[69] U.S. Department of Energy, Data Center Assessment Tool Suite "DC Pro", Master List of Energy Efficiency Actions, Save Energy Now, 2008, http://www1.eere.energy.gov/industry/datacenters/pdfs/data_center_actions_list.pdf (accessed 2011-03-22).
[70] P. Mahesh, Factsheet 33 on Green Computer: Making of a Greener Computer, New Delhi, 2009, http://toxicslink.org/dn.php?section=1&id=213&atn=0 (accessed 2011-03-23).
[71] J.L. Zapico, M. Turpeinen, N. Brandt, Greenalytics: a tool for mash-up life cycle assessment of websites, in: K. Greve, A.B. Cremers (Eds.), EnviroInfo 2010, Integration of Environmental Information in Europe; Proceedings of the 24th International Conference on Informatics for Environmental Protection, October 6–8, 2010, Cologne/Bonn, Germany, Shaker, Aachen, 2010, pp. 754–763.

Stefan Naumann studied computer science at the Universities of Kaiserslautern and Saarbrücken (Germany). He received his doctorate in natural sciences (Dr. rer. nat.) from the University of Hamburg in 2006. Since 2008 he is a full professor for fundamentals in computer science and mathematics, as well as environmental and sustainability informatics at the Environmental Campus Birkenfeld, a subsidiary of the Trier University of Applied Sciences. Here, he is a directorate member of the Institute for Software Systems in Business, Environment, and Administration. His research interests are sustainable development in conjunction with online communities and the environmental and social impacts of information technology, especially Green IT and Green by IT. Within this context, he investigates also questions on how non-professional IT users can be supported, so that they are able to participate successfully, e.g. in E-Government processes. Stefan Naumann has authored over 35 peer-reviewed articles in the context of environmental and sustainability informatics.

Markus Dick graduated in computer sciences (degree: Diplom-Informatiker (FH), equivalent to BSc) from the Trier University of Applied Sciences (Environmental Campus Birkenfeld) (Germany). He received his MA in andragogy from the University of Kaiserslautern (Germany) in 2010. Since 2009 he works as a research associate in the R&D project "Green Software Engineering". As part of his work, he has authored a couple of international peer-reviewed conference papers in this area. His personal research interests are web applications, agile software engineering, and sustainable development in the context of adult education.

Eva Kern is a MSc candidate in computer science and media at the Trier University of Applied Sciences (Environmental Campus Birkenfeld) (Germany). She received her BSc in computer sciences and media in 2010. Since 2010 she works as a research assistant in the R&D project "Green Software Engineering". Her research interests are the relevance of sustainability for personal life, the role of sustainability communication in different kinds of media and the potentials of computer sciences in the context of a sustainable way of living.

Timo Johann is a MSc candidate in applied computer sciences at the Trier University of Applied Sciences (Environmental Campus Birkenfeld) (Germany). He received his B.Sc. in computer science in 2008. From 2008 to 2010, he worked as a research assistant in the R&D project "e-Government and Environmental Science". Since 2010 he works as a research assistant in the R&D project "Green Software Engineering". His research interests are Sustainable Development and Life Cycle Assessment of Software Products.