


ISS Institute for Software Systems  
In Business, Environment and Administration

# Enhancing Software Engineering Processes towards Sustainable Software Product Design

Markus Dick, Stefan Naumann



Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

Markus Dick, Stefan Naumann  
{m.dick, s.naumann}@umwelt-campus.de


Trier University of Applied Sciences, Umwelt-Campus Birkenfeld  
Campusallee, D-55768 Hoppstädten-Weiersbach, Germany

<http://www.green-software-engineering.de/>

This presentation corresponds to the following paper:

Dick, Markus; Naumann, Stefan: **Enhancing Software Engineering Processes towards Sustainable Software Product Design**. In: Greve, Klaus; Cremers, Armin B.: EnviroInfo 2010. Integration of Environmental Information in Europe. Proceedings of the 24<sup>th</sup> International Conference on Informatics for Environmental Protection, Cologne/Bonn, Germany. Aachen: Shaker Verlag, 2010, pp. 706 - 715.


The project “Green Software Engineering” (GREENSOFT) 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.



ISS Institute for Software Systems  
In Business, Environment and Administration

## Motivation

- Power consumption of data centres in the world increased from 58 TW h in 2000 to 123 TW h in 2005
- Reducing the consumption of energy and natural resources caused by ICT is necessary
- Efforts exist in the field of computer hardware
- Lack of efforts in the field of computer software
- Up to now it is not clear what sustainable software or sustainable software engineering is



Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

2

The power consumption of data centres in the world increased from 58 TW h in 2000 to 123 TW h in 2005, and is still increasing.

Hence, reducing the consumption of energy and natural resources caused by ICT is necessary. Where manifold efforts exist in the field of computer hardware (that is: Green-IT), there is a lack of models, descriptions, or realizations in the field of computer software.

Especially, there are hardly any systematic methods available that try to integrate sustainability aspects into software product design and development, as it is common today for material products like cars, light bulbs or computer hardware. Furthermore, well-known textbooks on software engineering like Sommerville or Balzert that are used in lectures do not even mention sustainability aspects of software.

Additionally, up to now it is not clear what the terms “sustainable software” and “sustainable software engineering” mean or what they are.



## Outline


- I. Definitions (Sustainable Software, Sustainable SE)
- II. Life Cycle Thinking for Software Products
- III. A Generic Model for Sustainable Software Engineering
- IV. Instantiating the Model
- V. Summary & Outlook





## I. Definitions






ISS Institute for Software Systems  
In Business, Environment and Administration

## “Sustainable Software”

“**Sustainable Software** is software

- whose direct and indirect negative impacts on economy, society, human beings, and environment
- that result from development, deployment, usage, and disposal of the software are minimal and/or
- which has a positive effect on sustainable development”



Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

5

In this and in the following definition, we understand direct impacts as energy and resource demand that is necessary to “produce” use and dispose of the software product.

Indirect impacts are effects that result from using the software product on other processes, and long term systemic effects resulting from software usage.

Development, deployment, usage, and disposal address the whole lifecycle of a software product in analogy to ordinary “non-virtual” product lifecycles.



## “Sustainable Software Engineering”

“***Sustainable Software Engineering*** is the art of


- defining and developing software products in a way so that
- negative and positive impacts on sustainability that result or are expected to result from the software product
- over its whole lifecycle
- are continuously assessed, documented and optimized”






## **II. Life Cycle Thinking for Software Products**






ISS Institute for Software Systems  
In Business, Environment and Administration

## Life Cycle Thinking for Software Products



```
graph LR; A[Product Definition] --> B[Development]; B --> C[Distribution]; C --> D[Acquisition]; D --> E[Usage]; E --> F[Deactivation]; F --> G[Disposal];
```




Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

8

If you look at the phases of our lifecycle model, you will recognize that it is more a product life cycle in the sense of Life Cycle Thinking (also attributed as a “cradle-to-grave approach”) than an ordinary software life cycle or software development process, because these are usually focusing on development phases and development activities.

This model mainly fits standard software products. For custom software products the phase “Acquisition” follows close upon the phase “Product Definition”.






ISS Institute for Software Systems  
In Business, Environment and Administration


## Life Cycle Thinking for Software Products

Sustainability relevant criteria



```
graph LR; A[Product Definition] --> B[Development]; B --> C[Distribution]; C --> D[Acquisition]; D --> E[Usage]; E --> F[Deactivation]; F --> G[Disposal];
```

Product Definition Development Distribution Acquisition Usage Deactivation Disposal




Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

9

This lifecycle model has two objectives:

Its first objective is to assign criteria to the different lifecycle phases that lead to or result in sustainability relevant effects.




ISS Institute for Software Systems  
In Business, Environment and Administration

## Life Cycle Thinking for Software Products

Sustainability relevant criteria

Product Definition Development Distribution Acquisition Usage Deactivation Disposal

Starting points for activities




Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

10

Its second objective is to provide starting points for activities that hopefully lead to more sustainable software products.


Now, let us have a look on some example criteria.




ISS Institute for Software Systems  
In Business, Environment and Administration

## Sustainability Relevant Criteria

- ...
- Transportation for daily way to work
- Working conditions (offshore workers)
- Business trips
- Energy for ICT
- Office lighting
- Office HVAC



Product Definition → Development → Distribution → Acquisition → Usage → Deactivation → Disposal




Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

11

Please, note that these examples are far from complete.

Appropriate criteria for the development phase are:

- Working Conditions of offshore workers
- Business trips for meetings with the development team
- Energy for the necessary IT infrastructure
- Office heating and air conditioning



## Sustainability Relevant Criteria

- ...
- Manuals
- Transportation
- Packaging
- Data medium
- Download size

- ...
- Manuals
- Data medium
- Packaging

Product Definition

Development


Distribution

Acquisition

Usage

Deactivation

Disposal



Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER


12

Appropriate criteria for the distribution phase are

- Printed manuals
- Packaging
- Data medium
- Download size (if you software product is offered as a download)

Some of these relate directly to criteria of the disposal phase, like


- The disposal of printed manuals
- data mediums
- packaging




**ISS** Institute for Software Systems  
In Business, Environment and Administration

## Sustainability Relevant Criteria

- ...
- Accessibility
- Update size & frequency
- Screen size requirements
- Hardware requirements
- Memory & processor usage
- Network load

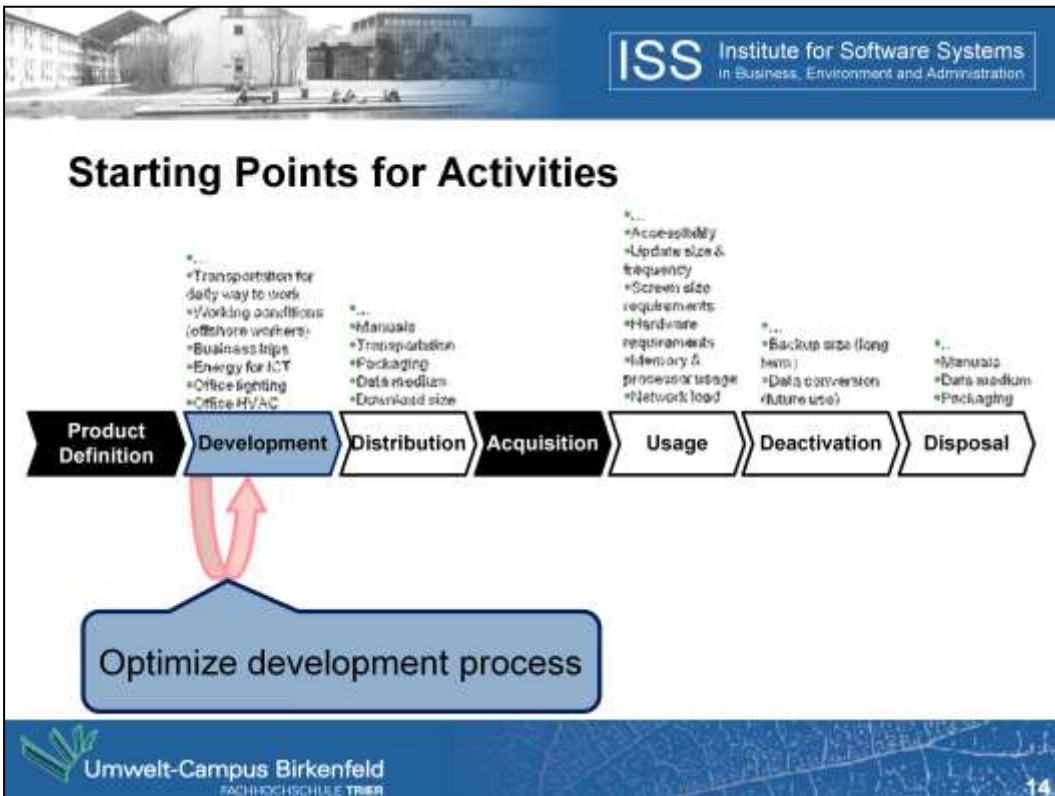


 Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

13

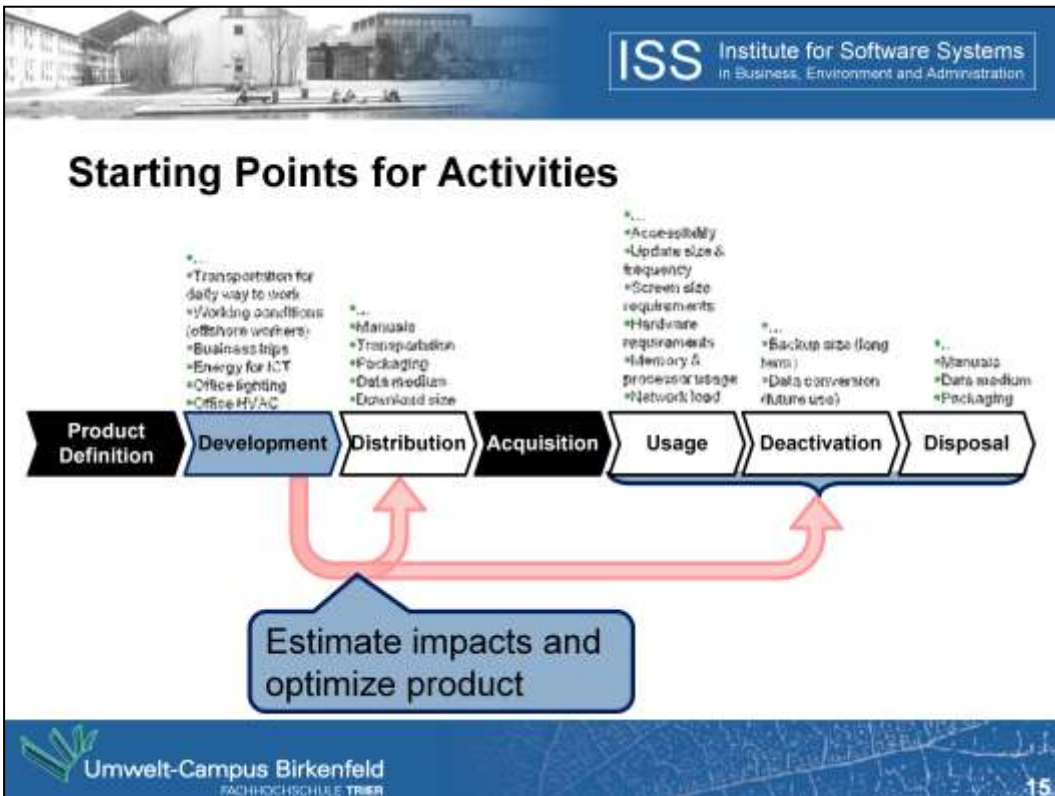
Examples of criteria for the usage phase are:

- Accessibility issues
- Screen size requirements
- Hardware requirements
- Memory and processor usage during program execution



We are focusing on two starting points:

First, according to our given definitions, the software development process itself should be optimized in order to mitigate negative impacts or to enforce positive impacts that result from it.



Second, during development, the positive or negative impacts that are expected to arise from distribution and future use of the software product should be continuously anticipated, assessed and reflected on by involved actors.

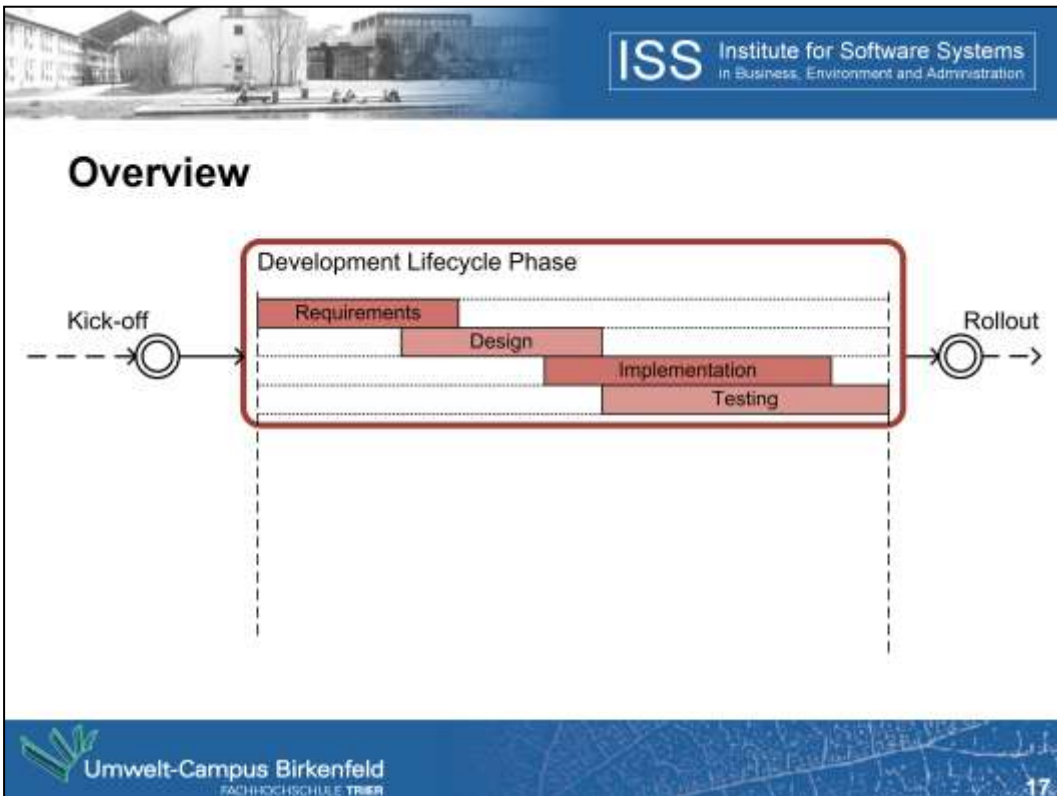
The outcomes of these assessments and reflections should then be used to take action towards a more sustainable software product.



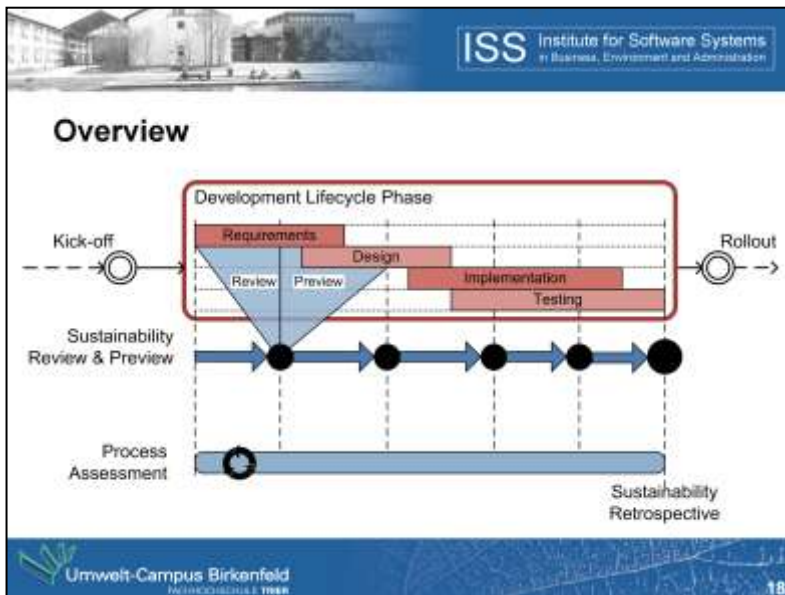
### **III. A Generic Model for Sustainable Software Engineering**







For this introductory overview, we are using a simple “waterfall-like” software development process with the exception that the different development phases are executed in parallel rather than strictly sequential. This is the more general case and enables us to subsume processes like the Unified Process with this model.



This general process is enhanced by several activities that have the objective to enable sustainable software engineering. These activities are: Sustainability Reviews & Previews, Process Assessment, and the Sustainability Retrospective

Sustainability Reviews & Previews mainly consider impacts on sustainability which are expected to arise from distribution and future use of the software product. In this activity, actors take a look at the work done and assess outcomes according to sustainability criteria. This is the review part.

As the preview part, actors develop and realize measures until the next Review & Preview in order to optimize the sustainability of the software product.

Sustainability Review & Previews take place after one-half or two-thirds of a process phase. This enables actors to realize software design or implementation alternatives within the same phase.

Depending on the length of a phase, it may be necessary to perform multiple Reviews & Previews. Reviews & Previews are a team facilitation approach. Involved actors review and assess their artefacts (e.g. requirements, architecture, coding) and develop and assess alternative solutions in order to choose the better “more sustainable” solution.

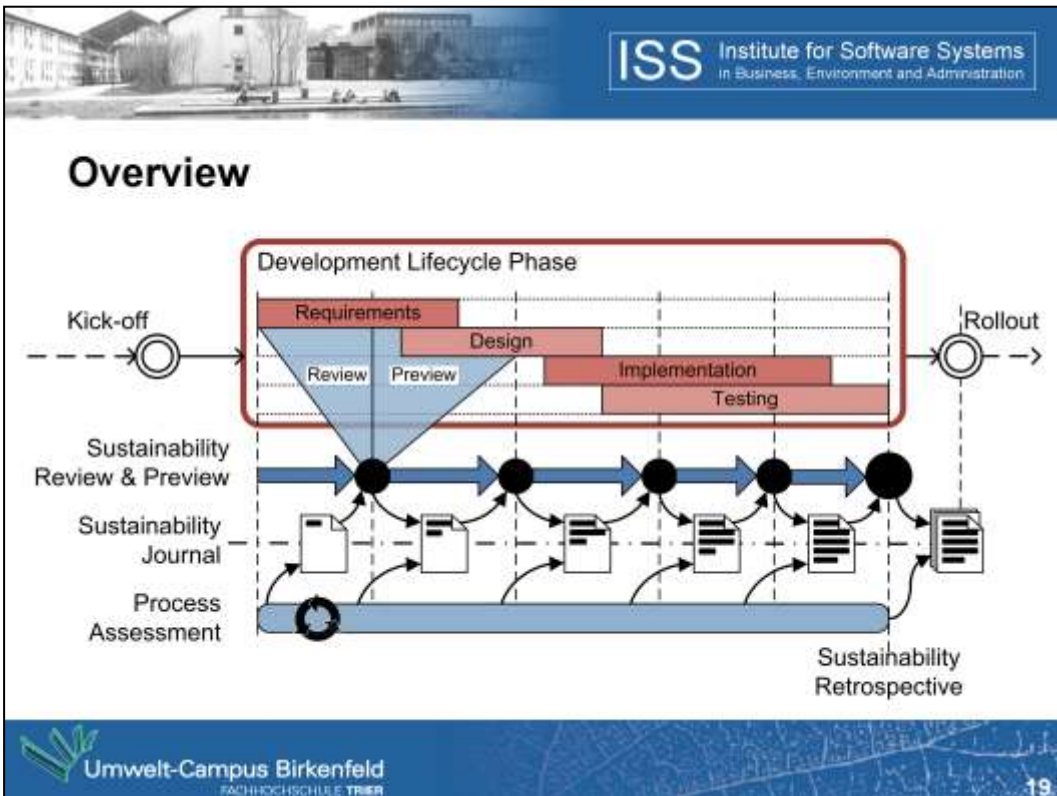
The continuous Process Assessment activity quantifies and assesses impacts on sustainability, which result from the software development process itself. Criteria of the Development Lifecycle Phase e.g. transportation for daily way to work, working conditions (offshore workers), business trips, energy for ICT, office HVAC, pro rata impacts of common corporate departments.

Thus, Sustainability Reviews & Previews and Process Assessment covers our two starting points of activities, which I mentioned earlier.

At the end of the development process, the Sustainability Retrospective combines the results of Reviews & Previews (mainly impacts that are expected from the usage phase) and Process Assessments (mainly impacts that result from the development phase). Thus it covers impacts over the whole lifecycle of the software product. The outcomes should be reported to stakeholders of the software product.

Additionally, it looks for ways to improve upcoming software development projects and their resulting software products in some kind of a team learning approach.

The expected outcomes of this learning approach can be e.g. decisions for future projects, lessons learned, best practices regarding sustainability issues of software products or development processes.



The Sustainability Journal is the information hub of our process enhancements. It is a well structured report, which evolves simultaneously with the software project. Its purpose is to document Sustainability Reviews & Previews, Process Assessment and the Sustainability Retrospective. Finally, after the project has finished, it reports the assessed impacts on sustainability.



## Tools, Guidance, and Educational Material

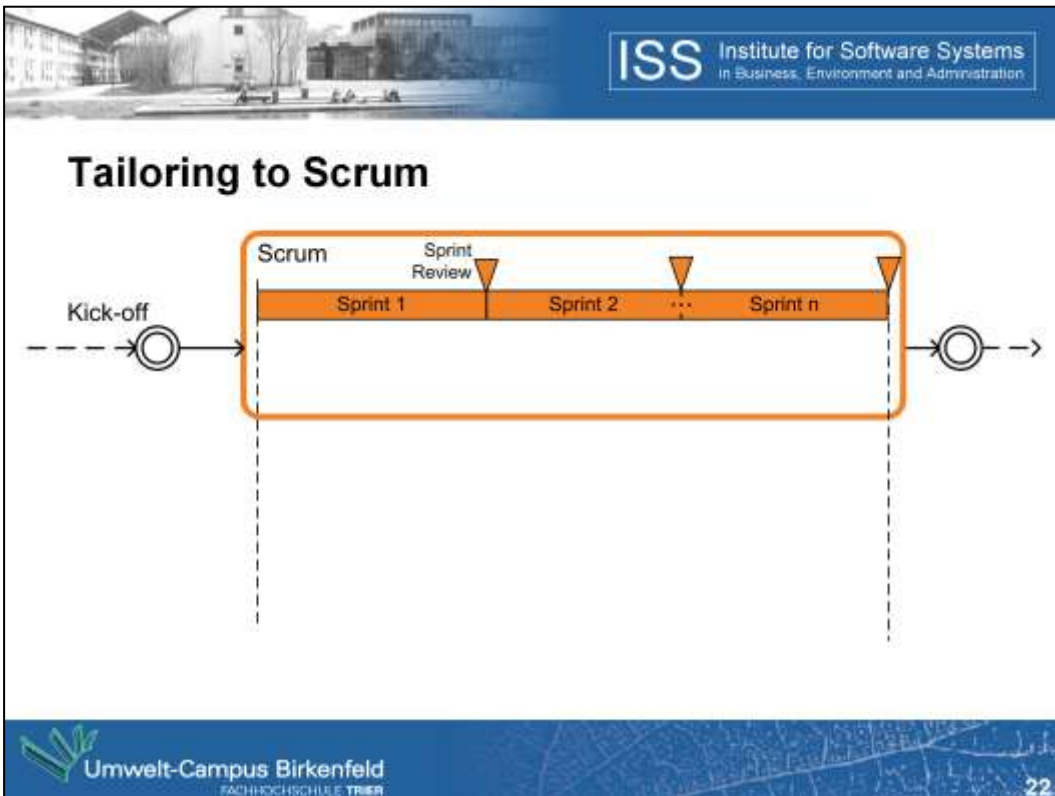
- Proposed process enhancements are supported by tools
  - Support collection of data during process assessment
  - Assist actors with product lifecycle issues/criteria
- Actors need recommendations for action (e.g. tips, guidelines, checklists) to refine e.g. architecture and coding in order to mitigate impacts on sustainability
- Sustainability issues are not well known to computer scientists → Educational material is necessary





## **IV. Instantiating the Model**





Now I will show, how we tailored our proposed enhancements so that it fits a non-waterfall-like approach.

For this purpose, we chose Scrum, because it is quite different from the waterfall-like-approach that I used initially to show how our enhancements work in principle.

Scrum is a “low ceremony” and agile software development process, which was developed by Ken Schwaber and others.

With Scrum, a piece of software is developed by several month-long iterations, so called Sprints.

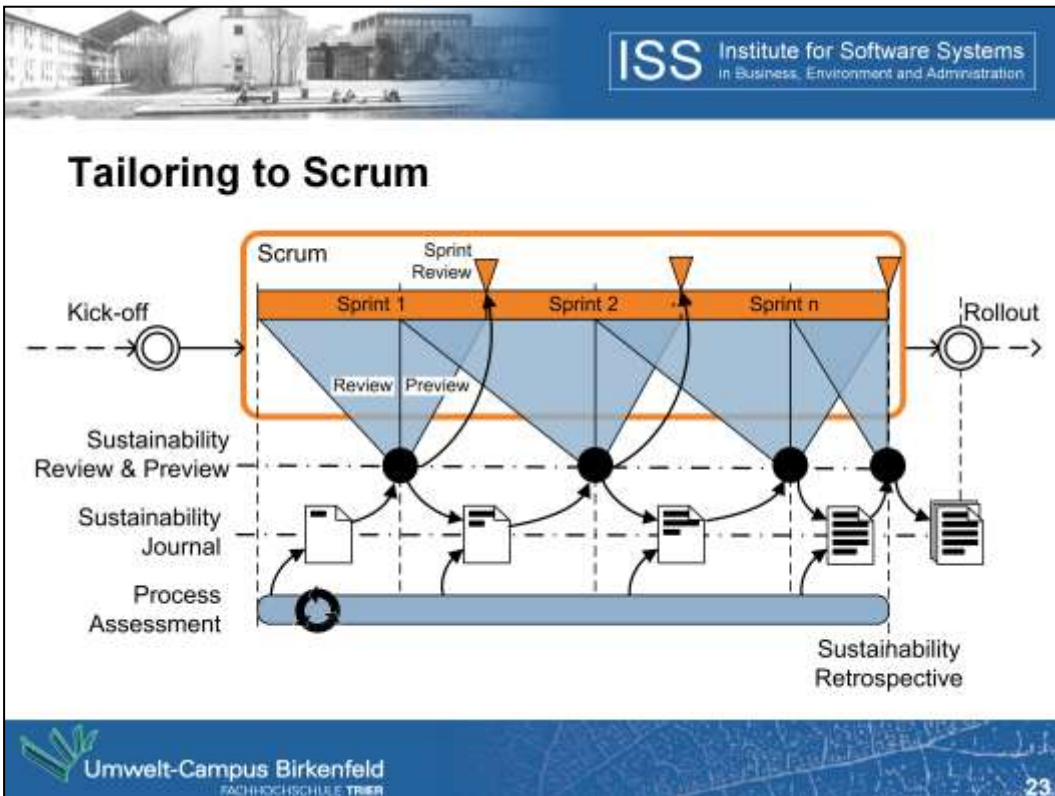
Each Sprint delivers a potentially shippable software product increment.

Besides these Sprints, Scrum does not define special development phases or activities, like design, implementation, testing, etc.

Such activities occur implicitly within a sprint, just when they are needed by developers.

Each Sprint ends with a so called Sprint Review. Here, the product increment is presented to the Product Owner, customers, management, etc.


The Product Owner is a representative of all stakeholders and he accepts or rejects the work results of the recent Sprint.



According to our proposal, Sustainability Reviews & Previews should take place after two-thirds of a Sprint. This enables the development team to implement more sustainable alternatives within the current sprint and to deliver an potentially shippable product increment at the end of the Sprint. The outcomes of the Reviews & Previews should be reported to the stakeholders during Sprint Reviews.


The Sustainability Retrospective should take place just before the end of the last Sprint. Thus, the development team is able to report the combined assessment results to the stakeholders in the final Sprint Review meeting. After the project has finished, the team should discuss the other aspects of the Sustainability Retrospective, like e.g. decisions for future projects, lessons learned, or best practices regarding sustainability issues. This ensures that the team can discuss and reflect on these aspects without pressure, which hopefully leads to better results.





## Early Results

- Decisions on SW architecture based upon memory and consumed processing time are difficult to prepare
  - Appropriate tools are necessary (e.g. profilers, performance test tools)
  - Results depend on implementation details of used APIs and libraries (reactions are usually not predictable)
  - Elaborate performance tests are necessary
- Test results and decisions should be stored for reference and reuse → Knowledge Base



As a first step, we applied our process enhancements to a small one-month Scrum-driven student software project with 3 participants. This project had 4 one-week Sprints. Of course, these Sprints were too short, but during these Sprints, the students accomplished 3 Sustainability Reviews & Previews and 1 Sustainability Retrospective.

It came out that decisions on software architecture based upon memory and consumed processing time according to our sustainability criteria are difficult to prepare.

This has several reasons:

- Developers need appropriate tools like e.g. profilers and performance test tools, especially when they use programming languages that automatically handle memory allocation.
- The results of these tools depend on implementation details of the used APIs, frameworks, and libraries. If you change e.g. an implementation of an API, then the reactions regarding memory and CPU usage are usually not predictable.
- Hence, elaborate performance tests that simulate the expected real-life CPU and memory load are necessary.

Preparing decisions on software architecture, which are based upon several design or implementation alternatives can be time consuming and therefore expensive.


Hence, the results and their decisions should be stored for reference and reuse in future software projects.





## V. Summary & Outlook






ISS Institute for Software Systems  
In Business, Environment and Administration

## Summary

- It is not clear whether energy savings through ICT outbalances energy consumption or not
- It is rational to integrate sustainability aspects not only in hardware products, but also in software products
- We presented
  - A Lifecycle model for software products
  - A generic process model for Sustainable Software Engineering
  - An example how this model can be applied to other software processes



Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER


26

Summarizing, it is currently not clear whether energy savings through information and communication technology outbalances its energy consumption or not.

In either case it is rational to integrate sustainability aspects into software product design and development as it is already common today for material products, like cars, light bulbs or computer hardware.

Hence, we presented


- a life cycle thinking inspired life cycle model for software products,
- a generic process model for Sustainable Software Engineering that can be tailored to fit arbitrary software development process models
- An example that shows how our generic process model can be tailored to a non-waterfall-like but agile software process



ISS Institute for Software Systems  
In Business, Environment and Administration

## Outlook

- Detail and broaden our lifecycle model
  - Criteria for different software scenarios and software types
  - Criteria addressing social and economic dimensions
  - Indirect effects (examples and educational material)
- Define how process assessment works in detail
- Tailor and Evaluate our proposed generic process enhancement in real life software projects
- Develop a knowledge base to support sustainable software development, administration, and use




Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

27

Our next steps are to detail and broaden our model with e.g.

- More criteria for different software scenarios and types of software
- Criteria that addresses the social and economic dimensions of sustainability
- Examples and educational material that address indirect effects of software use, because we do not expect software developers to recognize these intuitively.

We plan to examine, how Process Assessment works in detail, we plan to tailor an evaluate our enhancements in real life software projects, and we plan to develop and operate the already mentioned knowledge base that supports development, administration, and use of software in a more sustainable way.



**ISS** Institute for Software Systems  
In Business, Environment and Administration

**Thank you for your attention!**


Feel free to contact us:

Markus Dick      m.dick@umwelt-campus.de

Trier University of Applied Sciences  
Environmental Campus Birkenfeld  
Institute for Software Systems  
Germany


[greensoft@umwelt-campus.de](mailto:greensoft@umwelt-campus.de)  
<http://www.green-software-engineering.de/>

SPONSORED BY THE



Federal Ministry  
of Education  
and Research

Ref.-No. 17N1209



Umwelt-Campus Birkenfeld  
FACHHOCHSCHULE TRIER

28

The project “Green Software Engineering” (GREENSOFT) 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.