Volume 4, No. 1, January 2013
Journal of Global Research in Computer Science
ISSN-2229-371X
REVIEW ARTICLE
Available Online at www.jgrcs.info

GREEN SOFTWARE ENGINEERING PROCESS : MOVING TOWARDS SUSTAINABLE SOFTWARE PRODUCT DESIGN

Shantanu Ray *[1], Nabaraj Sengupta [2], Koustav Maitra [3], Kaushik Goswami [4], Shalabh Agarwal [5] and Asoke Nath[6]
*[1] Computer Science,St.Xavier's College(Autonomous),Kolkata,West Bengal,India
shantanuray.123@gmail.com
[2] Computer Science,St.Xavier's College(Autonomous),Kolkata,West Bengal,India
nabaraj.sengupta@gmail.com
[3] Computer Science,St.Xavier's College(Autonomous),Kolkata,West Bengal,India
koustav.sxc@gmail.com
[4] Computer Science,St.Xavier's College(Autonomous),Kolkata,West Bengal,India
goswamikaushik@hotmail.com
[5]Computer Science,St.Xavier's College(Autonomous),Kolkata,West Bengal,India
shalabh@sxccal.edu
[6]Computer Science,St.Xavier's College(Autonomous),Kolkata,West Bengal,India
asokejoy1@gmail.com

Abstract: The Software development lifecycle (SDLC) currently focuses on systematic execution and maintenance of software by dividing the software development process into various phases that include requirements-gathering, design, implementation, testing, deployment and maintenance. The problem here is that certain important decisions taken in these phases like use of paper, generation of e-Waste, power consumption and increased carbon foot print by means of travel, Air-conditioning etc may harm the environment directly or indirectly. There is a dearth of models that define how a software can be developed and maintained in an environment friendly way. This paper discusses the changes in the existing SDLC and suggests appropriate steps which can lead to lower carbon emissions, power and paper use, thus helping organizations to move towards greener and sustainable software development.

Keywords: [SDLC, design, maintenance, carbon emissions, e-Waste]

INTRODUCTION

It has been found that ICT is responsible for 2% of total world emissions [1]. In fact, this 2% includes the in-use phase of software and remaining 98% software is operational in private sector during its business and also in public sector while supporting society. Software can become greener at least two ways. First, by being much more energy efficient, and henceforth using lesser resources and causing fewer CO2 emissions. Second, by making those processes sustainable which it is supporting which is mainly decreasing the emissions of governments, companies and also individuals. So enterprise software's must keep in mind the sustainability issues and they must support sustainable business models which are innovative. GREEN 2012 came up with a special theme 'Green Knowledge for Sustainable Software Engineering.' keeping in mind the concept of green software.

It brings together software engineering researchers and practitioners to discuss the state-of-the-art and state-of-the-practice in green software, as well as research challenges, novel ideas, methods, experiences, and tools to support the engineering of sustainable and energy efficient software systems. Maintaining and enhancing of the software is much more difficult than the initial development phase. According to Sun Microsystems Java coding conventions, 80% of software lifecycle costs are for maintenance. According to the same source, original authors wont like to complete the whole task. So in conclusion the common best practices can lead to an ease in cost. Even in server and mobile applications the importance of software life cycle costs can be emphasized. In such application domains we see the long product lifecycles and many supported technology platforms and environments can also be there. Sustainable software methodology has gained quality improvements and has a great cost saving nature for which it is highly appreciated. The software architecture is still more manageable, and most importantly costs can be predictable, and quality assurance is straightforward in this case. SCRUM and EXTREME PROGRAMMING (XP) are so called agile methods which are being used by many software projects. Agile methods helps in minimizing the inefficiency and inflexibility of more rigid development models, and they often take the customer as a part of the project.

The project is divided by the agile method into sprints with duration of one or two week. Each sprint produces a working, testable part. Customer continuously gives the feedback and hence the next sprints work-plan is adjusted accordingly. In our view, projects with no fixed requirements lists or the lists with lot of unknowns are the ones where agile methods shine, and there is a small project team and where customer actively participates in the project.

Sustainable software:

Sustainable software is defined as the 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[2]**.** In this 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.

The maintenance activities can be shortened as a result of the software which can be used for a longer period of time without any modification and thus will help minimizing the carbon footprint[3]. Any maintenance activity of software is considered to be a fresh development effort and results in the repeat of the entire SDLC process. Hence software which has a longer lifetime can also be considered to be sustainable software. The term sustainable applies both the longer life and greener aspects of software.

The biggest technology that has enabled software-centric Green IT approach is virtualization. Organizations, both big and small are adapting virtualization mainly to cut down the cost of their servers and desktops. Inadvertently, this has also contributed a lot towards a sustainable software environment. Hence the way the software is used is important towards the sustainability of the software.

Green and Sustainable Software:

Green and Sustainable software is a software whose direct and indirect consumption of natural resources which arise out of deployment and utilization, are monitored, continuously measured, evaluated and optimized already in the development process, appropriation and utilization aftermath can be continuously evaluated and optimized, development and production processes cyclically evaluate and minimize their direct and indirect consumption of natural resources and energy [5]. There are some common practices in software development which are definitely not green. For example, advanced computer hardware and latest technology compensates for the shortcomings in the software product and hence slower, inefficient and more expensive software has been there for a long time without much of performance degradation. Software engineers do cost analysis in a way that the man hours are considered to be the most precious commodity and hence emphasis is to reduce the development time. Sustainable Software is software, whose impacts on economy, society, human beings, and environment that result from development, deployment, and usage of the software are minimal and/or which have a positive effect on sustainable development[4]. Sustainable Software Engineering is the art of defining and developing software products in a way so that the negative and positive impacts on sustainability that result and/or are expected to result from the software product over its whole lifecycle are continuously assessed, documented, and optimized

Life Cycle of Sustainable Product:

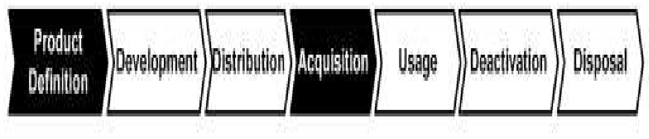


Figure-1

It is mainly a product life cycle in the sense of Life Cycle Thinking than an ordinary software life cycle or software development process due to its focus on development phases and development activities.

This model is designed for standard software products. For custom software products the phase "Product Definition" is immensely important.

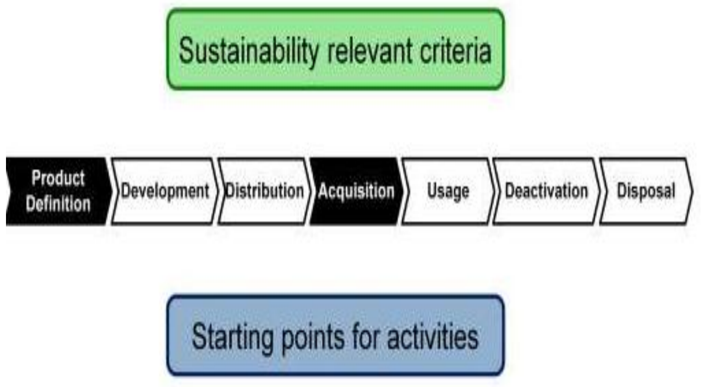


Figure-2

Appropriate criteria for the development phase are:
Working Conditions of offshore workers- the workers involved in the project should be well aware of the project specification and should be well equipped with work experience.

Business trips for meetings with the development team- the development team members should meet in the conferences to update themselves with the current market condition and business tricks.

Energy for the necessary IT infrastructure- the IT farms must be energy efficient to deal with the project and design specification.

Office heating and air conditioning- the offices may or may not be highly air conditioned.

Appropriate criteria for the distribution phase are:
Printed manuals- the manuals must have well printed description about the product and maintenance of the product.

Packaging- the packaging of the product may be attractive and be specific enough about the product of the software.

Data medium- the data transfer and the medium of the data must be simple enough for the unaware users of the product. Download size- the users must be aware of the download size for if the data is no longer accessible due to formatting, attack of viruses, etc.

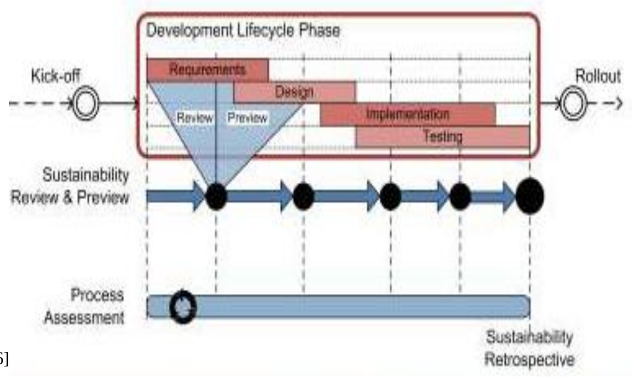
Examples of criteria for the usage phase are:
Accessibility issues- the hardware, tools and the software must be made available for accessing the module of the software.

Screen size requirements- the screen size requirement is of no importance until the software is being launched in its suitable environment.

Hardware requirements- the hardware requirements are chosen such that they should meet the requirements of the software and the software can have access to any environment.

A Generic Model for Sustainable Software Engineering:

Software life cycle describes the various phases in the life cycle of software. It describes the order in which the phases are executed. In the software life cycle each phase produces certain deliverables required by the next phase.


Development Lifecycle Phase
Kick-off
Requirements
Design
Review
Preview
Implementation
Testing
Rollout
Sustainability Review & Preview
Process Assessment
Sustainability Retrospective

[6]

Figure-3

The above figure describes the following SDLC phases:

Requirements: The goal of this phase is to understand the exact requirements of user to document them properly.

Design: The goal of this phase is to transform the requirement specification into a structure that is suitable for implementation in some programming language.

Implementation: During this phase the design is implemented in some coding language and testing each unit of the program.

Testing: In this phase the smaller and isolated modules are integrated together and the entire system is tested as a single unit which functions on the whole as a software.

Sustainability Reviews & Previews: It mainly considers impacts on sustainability which are expected to arise from distribution and future use of the software product. This is the review part.

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

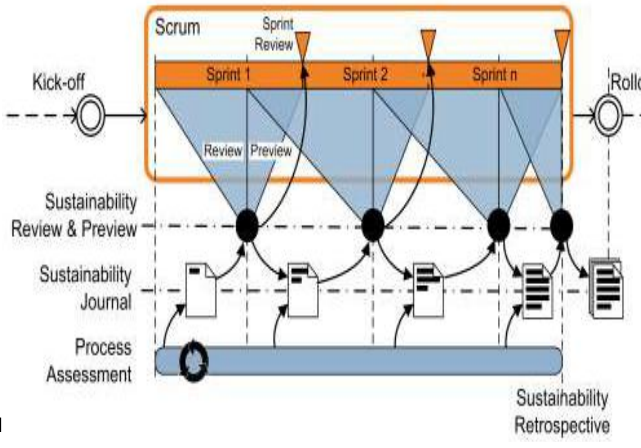
Sustainability Review & Previews take place after one-half or two-thirds of a process phase. Multiple reviews and previews may need to perform on the basis of the needs.

Process Assessment: It is the activity that quantifies and assesses impacts on sustainability, which result from the software development process itself.

A criterion of the Development Lifecycle Phase includes transportation for daily way to work, working conditions (offshore workers), business trips, energy for ICT.

Sustainability Retrospective: It covers the results of reviews, previews and process assessment.

Instantiating the model:



[6]

Figure-4

The above figure describes the following terms and processes:

Terms:
Scrum: It is defined as a “low ceremony” and agile software development process. But that does not mean that it will define all the software development process [1].

Sustainability Journal: It is the information hub of the process enhancement.

Sprint: Sprint is a piece of software developed by several month-long iterations. Each Sprint delivers a potentially shippable software product increment. Each Sprint ends with a so called Sprint Review.

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

Process:

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 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.

Early results:

a. [8]Decisions on SW architecture based upon memory and consumed processing time are difficult to prepare
    (a). Appropriate tools are necessary
    (b). Results depend on implementation details of used APIs and libraries
    (c). Elaborate performance necessary
b. [8]Test results and decisions should be stored for reference and reuse → Knowledge base

Case study-OCSA DAI:

[11]OCSA DAI can be sighted an example of sustainable software.OGSA-DAI was created which is an intuitive user interface and provided the software preconfigured on a virtual machine. OGSA-DAI is an innovative solution for distributed data access and management. It has been under development since 2002 and is now an established open source product currently managed by EPCC, The University of Edinburgh. OGSA-DAI allows data resources (e.g. relational or XML databases, files r web services) to be federated and accessed via web services on the web or within grids or clouds. Via these web services, data can be queried, updated, transformed and combined in various ways.

OGSA-DAI will contribute in futoure in which researchers and business users move away from technical issues such as data location, data structure, data transfer and the ins and outs of data integration and instead focus on application-specific data analysis and processing.

CONCLUSION AND FUTURE SCOPE

[7]Solid Works Sustainability technology enables us to quickly and easily assess the environmental impact of our design to create more sustainable products. The software integrates powerful Life Cycle Assessment-based tools into our established design process, measuring the environmental impacts of Carbon, Energy, Air and Water. The material selection tool provides instant feedback to help us choose the most environmentally-friendly material for a particular design. Analysis of parts, assemblies, and configurations, as well as automatic saving of environmental baselines, makes for better iterative design, while one-click report generation helps you easily communicate your findings.

ACKNOWLEDGEMENT

We are very much thankful to the faculties of the computer science department of St.Xavier's College, Kolkata for suggesting to make a study on the various topics of green software engineering and get acquainted with the terms of the green software.

REFERENCES

[1]. Green IT: Why Mid-size Companies are investingnow,Info-Tech Research Group,Toronto,Ontario M5E 1B3.

[2]. Fischer, J, Naumann, S & Dick, M 2010, 'Enhancing Sustainability of the Software Life Cycle via a Generic Knowledge Base' in EnviroInfo 2010.

[3]. International Organization for Standardization (ed.): ISO/IEC 25000:2005 Software Engineering—Software Product Quality Requirements and Evaluation (SQuaRE)—Guide to SQuaRE. International Organization for Standardization, Geneva (2005).

[4]. Markus Dick, Stefan NaumannTrier University of Applied Sciences, Umwelt-Campus Birkenfeld Campusallee, D-55768 Hoppstädten-Weiersbach Enhancing Software Engineering Processes towards Sustainable Software Product Design

[5]. Göhring, W.: The Memorandum "Sustainable Information Society". In: Proceedings 18th International Conference Informatics for Environmental Protection, EnviroInfo, Geneva (2004), http://www.wolf_gohering.de/OverMemoSIS.pdf

[6]. Dick, M, Naumann, S & Kuhn, N 2010, 'A Model and Selected Instances of Green and Sustainable Software'. Available from: http://www.green-software-engineering.de

[7]. Green It: The Global Benchmark, A Report On Sustainable It In The Usa, Uk, Australia And India,

[8]. International Organization for Standardization (ed.): ISO/IEC 12207:2008 Systems and software engineering - Software life cycle processes. International Organization for Standardization, Geneva (2008).

[9]. Going Green: A Holistic Approach to Transform Business - Dr. Sajal Kabiraj, Dr. Vinay Topkar, R.C. Walke

[10]. Hilty, L. M. (2005): Information systems for sustainable development, Idea Group Publishing, Hershey, Pa.

[11]. Accesing data using a common interface-OCSA DAI as an example-Elias Theocharopoulos and Tilaye Alemu ISSGC '09-Sophia Antipolis

Short Bio Data for the Author

Shantanu Ray is a student of computer science department in St.Xavier's College (Kolkata). His primary interest is in the field of web based technologies, green computing. He has developed an online evaluation system using HTML, PHP, and MySql.

Nabaraj Sengupta is a student of computer science department in St.Xavier's College (Kolkata). His primary interest is in the field of web based technologies, green computing. He has developed an online evaluation system using HTML,PHP,MySql.

Koustav Maitra is a student of computer science department in St.Xavier's College(Kolkata). He is working on a project using MATLAB, image processing.

Professor Kaushik Goswami is an assistant professor in computer science department. He is busy with different projects. His primary research area is UNIX, internet technologies.

Professor Shalabh Agarwal is an assistant professor in computer science department. He is busy with different research projects. His primary research area is green computing, e-learning, software engineering. He has published several papers in national and international journals and conference proceedings.

Asoke Nath is the associate professor in department of computer science. Apart from his teaching assignment he is involved with various research works in cryptography, steganography, green computing, e-learning.
He has presented papers and invited tutorials in different international and national conferences in India and abroad.