A Methodology to Derive Sustainability Indicators for Software Development Projects

Giuseppe Lami
CNR - ISTI
Via G. Moruzzi, 1
I-56124 Pisa, Italy
+39 0503153493
giuseppe.lami@isti.cnr.it

Fabrizio Fabbrini
CNR - ISTI
Via G. Moruzzi, 1
I-56124 Pisa, Italy
+39 0503153493
fabrizio.fabbrini@isti.cnr.it

Mario Fusani
CNR - ISTI
Via G. Moruzzi, 1
I-56124 Pisa, Italy
+39 0503153493
mario.fusani@isti.cnr.it

ABSTRACT
The ever-increasing pervasiveness of Information and Communication Technologies (ICT) not only is determining radical changes in everyone's life-style, in the social organizations, and in the economic dynamics, but it is causing relevant direct and indirect effects on the environment as well. One of the aspects not yet sufficiently addressed in literature is the sustainability of the software lifecycle. In this paper, we focus on the sustainability management of software development projects. In the practice, software development projects are managed by considering principally parameters as time, costs and quality of work products. We propose a methodological approach aimed at identifying sustainability indicators to be used in project management to set up sustainability objectives for a software development project, to control their achievement during the project itself and, possibly, to adopt corrective actions to maintain the project aligned with them. The availability of such sustainability indicators may contribute to reduce the carbon footprint of the activities performed in software development projects as well as to speed-up the development of a green culture in organizations developing software.

Categories and Subject Descriptors
D.2.8 [**Software Engineering**] Metrics.

D.2.9 [**Software Engineering**] Management – *Lifecycle*

K.6.1 [**Management of Computing and Information Systems**] Project and People Management

General Terms
Management, Measurement.

Keywords
Sustainability, Software Project Management, Goal-Question-Metrics.

1. INTRODUCTION
The preservation of the environmental resources and the sustainable development are among the most important objectives human beings will have to achieve in the next decades.

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
ICSSP' 13, May 18–19, 2013, San Francisco, CA, USA
Copyright 2013 ACM 978-1-4503-2062-7/13/05 ...$15.00

The term 'sustainability' (originating from the Latin word *sustinēre* that means: "to hold up") shows the capability of enduring over time. In the last years sustainability has been more and more intended as the responsible use of the earth resources in order to "meet the needs of the present without compromising the ability of future generations to meet their own needs" [1]. Because software intensive systems and applications are more and more pervasive in all the activities of everyone's day life, the environmental impact of any aspect related to them has indeed become an issue [18, 19]. ICT sustainability can be addressed from different perspectives, some focused on the system efficiency in terms of power consumption, some on software algorithmic or architectural optimization, some others on effective and managed software engineering practices and tools, and more recently, on a process-centric approach to software sustainability.

The main focus of this paper is on the derivation of sustainability indicators to be used to verify whether specific sustainability objectives have been met in the activities related to the development of software and in the software product itself. In fact, the carbon footprint due to the development and management of all the aspects and phases related to the software lifecycle may vary significantly depending on the capability to manage sustainability issues in software development projects. While practices for the management of software development projects in terms of timing, costs and product quality are widely applied (mainly in mature organizations), little attention is paid to sustainability management in software development projects. For this reason, we aim at contributing to fill such a gap by providing a methodology to systematically derive quantitative sustainability objectives to be set-up, measured and monitored in software development projects.

In this paper we address the need of establishing software sustainability indicators in order to make it possible to include sustainability objectives among the quality objectives, technical constraints and project goals in software development as well as their verification and management.

The paper is structured as follows: in section II an overview of the existing approaches to software sustainability in the literature and in the practice are discussed. In section III the importance of the availability of sustainability indicators (i.e. tangible elements to be used as sources of information to evaluate software sustainability) is discussed. In Section IV a methodology to derive significant sustainability indicators for software projects management is presented and in the subsequent section V a sample of sustainability indicators is provided. Finally, in Section

VI conclusions are provided along with the indication of our future research directions.

2. RELATED STUDIES

ICT sustainability has been one of the hottest topics addressed by researchers during last years. ICT sustainability has been addressed from the following different perspectives:

Hardware energy efficiency: the efforts made according to this approach aim at designing and developing system components able to minimize the power consumption during their usage, as well as at using materials having a reduced environment impact during production and disposal. In fact, for some ICT products (such as servers or set-top boxes) it is essential to reduce the power consumption during use, because it comprises the largest share in their total life cycle impact; for others it is more important to optimize their design for recyclability or to avoid negative effects during end-of-life treatment [2, 3, 4, 5]. These issues are addressed by the ISO 14000 family of standards related to environmental management that aims at supporting organizations to minimize how their operations negatively affect the environment (i.e. cause adverse changes to air, water, or land), to comply with applicable laws, regulations, and other environmentally oriented requirements, and to continually improve in the above aspects. [6, 7]

Optimization of Algorithms: the objective is to find and adopt optimized algorithmic solutions to reduce both the direct carbon footprint of software (e.g. power consumption due to the CPU cycles) and the indirect effects on sustainability (i.e. the effects depending on the usage of the system in the specific operational context). Although software doesn't consume energy, it deeply affects the consumption of hardware equipment. That's why software is indirectly responsible for energy consumption. [8, 9]

Sustainable software processes: the basic consideration in this case is that the environmental impact of software may depend on the processes adopted for developing, managing, operating and maintaining the software itself. As a process determines the organizational *modus operandi* in a specific area, then different processes are responsible of different environmental impacts. From this perspective, a sustainable software process is a software process that meets its (realistic) sustainability objectives, expressed in terms of direct and indirect impacts on economy, society, human beings, and environment that result from its definition and deployment. The sustainability of the software process may significantly vary according to the way the interrelated activities composing the whole software process are performed. Such an approach to software sustainability has been recently addressed in [10, 11].

Green software engineering: the focus, in this case, is on the development phase of the software lifecycle, and on the adoption of principles and techniques aiming at optimizing sustainability. In particular, good techniques for software design, coding and V&V (Verification & Validation) can reduce the environmental impact during the development and during the operation and maintenance of the software [12, 13, 14]. In particular, the potential benefits due to advanced computer hardware are not fully exploited because of the use of slower, inefficient and more expensive software. Better hardware components often compensate worst software. [15]

In the following of this paper, we address the ICT sustainability from a different viewpoint: the sustainability management of software development projects. Starting from the sustainable software processes perspective, and, with the aim of making such an approach concrete and applicable in practice, we provide a methodology to evaluate the level of achievement of sustainability objectives in a systematic and sound way by means of concrete indicators.

3. SUSTAINABILITY MANAGEMENT IN SOFTWARE PROJECTS

In the authors' experience, the management of a software project is almost uniquely focused on such attributes as effort, cost, schedule and product quality (principally in terms of defects found in work products). As a confirmation of that, the most popular software process maturity schemes as the CMMI [16] and the ISO/IEC 15504 [17] standard are strongly oriented to suggest that kind of attributes in project management. In particular, the Quantitative Project Management (QPM) Key Process Area of the CMMI scheme provides two kinds of project attributes to be taken into account in order to define the project objectives to be achieved: quality attributes and process performance attributes. The examples provided in [16] lead the CMMI users to address effort, cost, schedule and product quality attributes in project management.

The ISO/IEC 15504 Part 5 provides an Exemplar process assessment model that includes examples of assessment indicators. This part of the standard is, in the practice, used as a reference for process definitions. Also the ISO/IEC 15504 Part 5, in the definition of the Project Management (called MAN.3) process and of its related process performance indicators provides exemplar project attributes. These attributes are: business and quality goals for the project, size and complexity of the project and project effort, schedule and budget.

Neither in CMMI nor in ISO/IEC 15504 sustainability attributes for software projects are mentioned.

Today, because more and more attention is devoted to the sustainability of ICT, also the sustainability management in software project needs to be taken into consideration and it represents an important research topic as well. [19]

A recent report issued in the framework of the "Saving the Climate @ the Speed of Light", an ETNO (European Telecommunications Network Operators' Association) and WWF (World Wildlife Fund) joint project for the sustainable use of ICT, [18] addresses the importance of ICT for the global sustainability. That report presents a roadmap for the production and use of ICT, aiming at reducing the gap between the academic studies on ICT and the actions of policymaking organizations devoted to put into practice initiatives for CO2 emissions reduction. In particular, in order to clarify the different ways ICT can contribute to the reduction of CO2 emissions and sustainable development, it identifies three levels of Effects of the ICT in terms of environmental impact: Direct, Indirect and Systemic Effects. The level of interest in this paper is the one related to Direct Effects. Direct Effects refer to those caused by ICT infrastructure and equipment, e.g. the resource consumption (including materials and energy) when producing ICT equipment, the energy consumption when using ICT, and the effects of the resulting electronic waste.

Many environmentally negative effects can be identified as belonging to the category of Direct Effects. The most relevant are:

- Greenhouse Gas Emissions
    - Power waste
    - Effort/Time waste

- Fuel waste
- Material waste production
- Paper waste

To make an organization able to manage such Direct Factors in software projects the first aspect to be considered is the organizational culture change. In fact, the awareness of ICT sustainability-related questions and the change of individuals' and organizations' behavior in favor of Green ICT practices is the principal aspect to be considered. The second aspect is to provide the organizations with effective means not only to determine precise, feasible and meaningful sustainability objectives but also to manage software projects according to them. In this paper we address the latter of these aspects.

Our approach is to define a methodology to provide organizations with a systematic and sound way to define concrete and significant sustainability indicators to be used in the management of software projects in order to make them greener.

A stepwise view of the sustainability management in software project is provided in Figure 1.

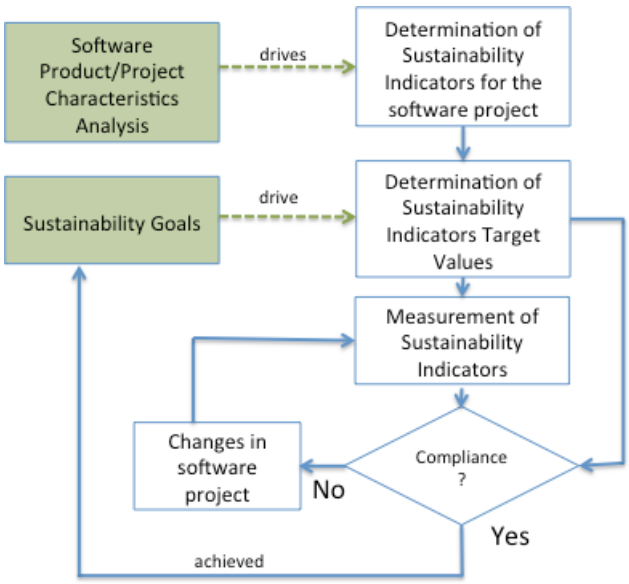

Figure 1. Sustainability management of software projects.

The first step is the determination, at the beginning of the project, of a set of meaningful sustainability indicators on the basis of the analysis of the characteristics of the project itself. Generally speaking, an indicator is a quantitative or qualitative variable able to give information that can be used to measure/evaluate the achievement of a defined goal. The availability of suitable indicators is fundamental to control that any activity, including software development projects, is respecting stated goals/objectives.

The second step is the definition of sustainability indicators' target values for the project (i.e. values to be achieved when sustainability indicators are measured/evaluated). Such a definition is driven by the Sustainability Goals to be defined according to the overall organization sustainability policy and the specific product characteristics and requirements.

The sustainability indicators are measured during the project deployment according to the project management procedures in order to control that their actual values are aligned with the related target values. The project should be adjusted if sustainability indicators target values are not achieved.

4. SUSTAINABILITY INDICATORS DEFINITION IN SOFTWARE PROJECTS

In this section a methodology, conceived by following a holistic approach, to define sustainability indicators for software projects is presented. Such a methodology is composed of two phases. The first phase aims at identifying those characteristics of the software development project (called Green Drivers) able to influence its environmental impact. In the second phase, the Green Drivers found in the first phase are used in a Goal-Question-Metrics-based mechanism to derive sustainability indicators for software projects.

4.1 A Bi-dimensional Scheme to Identify Green-Drivers

In the literature, many papers focusing on the identification (typically based on empirical studies or surveys) of the most relevant success factors for software projects exist.

A recent survey [20] considers most of these efforts and provides a summary of them. Thus, it represents an important reference on such a topic.

According to [20] the most important success factors can be grouped into four classes (called "dimensions" in [20]):

- People & Actions
- Development process
- Project Content
- Institutional Context

Each dimension is composed of a set of Factors as Table 1 shows. While these Success Factors have been derived without taking into account specifically sustainability purposes, nevertheless they are general enough to be considered valid for a wide range of purposes, including sustainability.
The pre-requisite for the definition of sustainability indicators to be used as basic elements in sustainability management of software projects is the identification of significant characteristics of the software projects themselves able to influence one or more Direct Effects (as defined in Section 3.). In the following we will refer to such characteristics (that can have technical or managerial nature) as *Green Drivers*.

Table 1. Software Projects' Success Factors classification

Success Factor Class	Success Factor
People & Actions	Developers
	Users
	Top management
	External agents
	Project team
	Social interaction
Project Content	Project characteristics
	Project scope, goals, objectives
	Resources
	Technology
Development processes	Requirements determination
	Project management
	Use of standard method
	User participation
	User training
	Management of change
Institutional Context	Organizational properties
	Environmental conditions

Green Drivers can be identified in different ways and according to different criteria. In order to make them rely on actually relevant aspects of the software project, the Success Factors indicated in Table 1 can be taken into account.
We propose, then, an approach based on a bi-dimensional scheme where the green-drivers can be derived by considering both the Success Factors and the Direct Effects dimensions. In Figure 2 such a scheme is depicted.

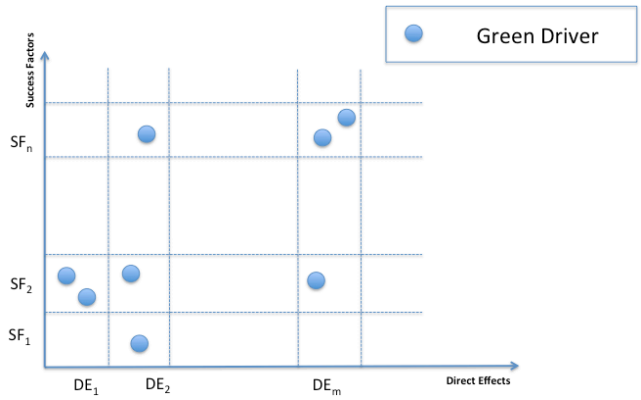

Figure 2. Bi-dimensional scheme for Green Drivers derivation.

In practice, for each Direct Effect, the Success Factors able to impact positively on it are selected and used to derive Green Drivers. The Green Drivers can be defined by the analysis of the way each specific Direct Effects can be improved by a Success Factor.

The schematic representation of such an approach is provided in Figure 3.

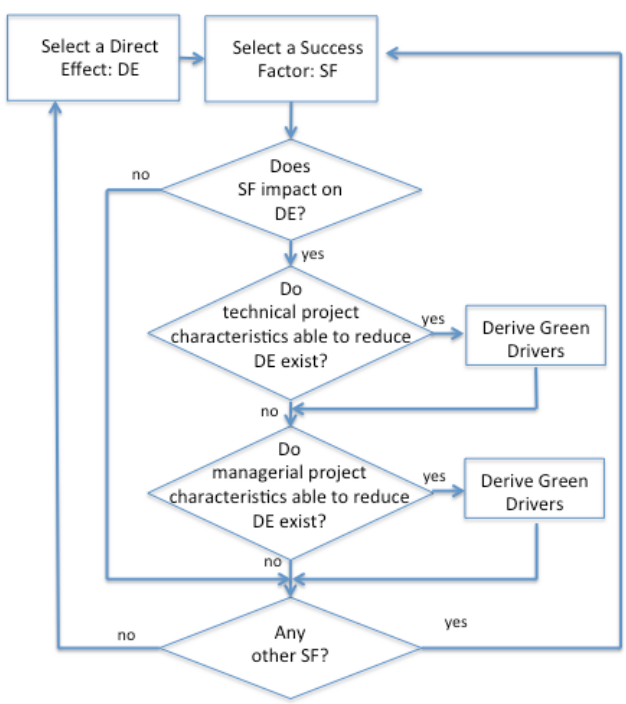

Figure 3. Green Drivers derivation procedure.

For example, according to the previous approach it is possible to realize that the Paper Consumption (Direct Effect) can be reduced by the Technology (i.e. according to Table 1., a Product Factor belonging to the Project Content dimension). A possible way to do that is the adoption of Dematerialization (i.e. the technology that allows to produce and distribute work products in electronic format). Then the use of technologies for Dematerialization is a project characteristic able to reduce paper consumption and can be identified as a Green Driver.

Such a scheme can be represented in tabular format too, as shown in Figure 4. According to this representation, the previous example would correspond to the following content of the cell having *Paper Consumption* as Direct Factor (column) and *Technology* (row) as Success Factor: Dematerialization.

Success Factors		Direct Effects				
		Power waste	Effort/Time waste	Material waste production	Paper waste	Fuel waste
People & Actions	Developers					
	Users					
	Top management					
	External agents					
	Project team					
	Social interaction					
Project content	Project characteristics					
	Project scope, goals, objectives					
	Resources					
	Technology					
nt processes	Requirements determination					
	Project management					
	Use of standard method					

Figure 4. Green Driver derivation table

4.2 A GQM-based Methodology to Define Sustainability Indicators

GQM (Goal-Question-Metrics) [21] is an approach to software metrics that defines a measurement model with a hierarchical structure. The starting level is represented by the goal (specifying purpose of measurement, object to be measured, issue to be measured, and viewpoint from which the measure is taken). The goal is refined into several questions that usually break down the issue into its major components. Each question is then refined into metrics, some of them objective, some others subjective. The same metric can be used in order to answer different questions under the same goal. Several GQM models can also have questions and metrics in common, making sure that, when the measure is actually taken, the different viewpoints are taken into account correctly (i.e., the metric might have different values when taken from different viewpoints).

The bi-dimensional scheme proposed in Section 4.1 can be used as a support to apply the GQM method with the aim of determining the relevant sustainability indicators for a software project. In the following of this sub-section, the general GQM method is tailored in order to make it suitable to the determination of sustainability indicators using the Green-Drivers described in section 4.1.

4.2.1 Goal Determination

According to the original GQM method the goals have to be derived by following the pre-defined template shown in Table 2.

Table 2. GQM goal definition template

Analyze	The object under measurement
For the purpose of	Understanding, controlling, or improving the object
With respect to	The quality focus of the object that the measurement focuses on
From the viewpoint of	The people that measure the object
In the context of	The environment in which measurement takes place

The derivation of goals can be made systematic by using the bi-dimensional scheme proposed in Section 4.1. In fact, the generic template in Table 2 can be made parametric by taking the information from the Green-Drivers table once it has been populated. The new template is shown in Table 3.

Table 3. Parametric GQM goal definition template

Analyze	*Green-Driver*$_{i,j}$
For the purpose of	`Improving`
With respect to	*Direct-Effect*$_i$
From the viewpoint of	`Project manager`
In the context of	`Software development project`

The modified goal definition template is composed of a variable part and a constant part.

The parametric part of the modified goal definition template is represented by:

- *Green-Driver*$_{i,j}$ *:* the element of the Green-Drivers table corresponding to the cell having as Direct-Effect the one on the ith row and as Success Factor the one corresponding to the jth column.
- *Direct-Effect*$_i$: the Direct-Effect corresponding to the ith row.

The other contents of the template represent its constant part.

4.2.2 Question Formulation

According to the original GQM approach, from the specification of each goal it is possible to derive meaningful questions that characterize that goal in a quantifiable matter. These questions can be classified into three groups. We provide a representation in parametric format of these three groups of questions in accordance with the Goal determination scheme defined in Section 4.2.1. Given a Goal $\mathcal{G}$ derived according to the template in Table 3, we indicate with $\mathcal{G}_\mathcal{GD}$ and $\mathcal{G}_\mathcal{DE}$ the specific Green Driver and Direct Effect used in $\mathcal{G}$ respectively.

Q1. How $\mathcal{G}_\mathcal{GD}$ can be characterized with respect the overall goal of $\mathcal{G}$?

Q2. How the attributes of the $\mathcal{G}_\mathcal{GD}$ that are relevant with respect to the $\mathcal{G}_\mathcal{DE}$ can be characterized?

Q3. How the characteristics of $\mathcal{G}_\mathcal{GD}$ that are relevant with respect to the $\mathcal{G}_\mathcal{DE}$ can be characterized?

4.2.3 Metrics Derivation

Once questions are defined, the final step of the general GQM model is the association of these questions with appropriate metrics able to give an answer to them. In our case the aim is to derive quantitative indicators (which will depend on the characteristics of the specific project) that allow answering the questions. In Section 5, exemplar indicators are derived for clarity.

5. AN EXAMPLAR SUSTAINABILITY INDICATOR DERIVATION

In this section, with the aim of clarifying the methodology described in the previous sections, an exemplar derivation of sustainability indicators for software projects is provided.

5.1 Phase 1: Green Drivers derivation

According to the indications provided in Section 4., the derivation of Green Drivers shall be based on the analysis of the characteristics of the project in order to identify the most significant ones, i.e. those able to influence one or more Direct Effects for each of the success factors.

The aim of this paper is to present a methodology to derive Indicators for the management of sustainability of software development projects. Thus, results of case studies or empirical experiences are not in the scope of this paper and they will be presented in further papers as soon as available. For this reason we do provide only some exemplar generic Green Drivers that have been obtained from an accurate analysis of the implications of the success factors on the sustainability of the project.

Table 4 contains an exemplar set of possible Green Drivers. The rational of these Green Drivers is provided in the following:

- *Development-supporting tools efficiency*: the extent to which the tools used for supporting the activities of software projects (e.g. requirements management, configuration management, coding environment, verification infrastructures...) are efficient (where the efficiency is intended as the power consumed vs. the services provided) may impact on the carbon footprint of the whole software development project.
- *Project Activities Value*: project activities having a little or no added value are responsible for waste of resources (as power or paper) and reduce the project sustainability.
- *Team members' skill:* project team composition influences the project sustainability because unskilled people may enlarge the duration of project activities as well as the time they spend at work (with consequent waste of resources as power, heat, fuel, paper, …).
- *Development-supporting tools suitability*: the extent to which the tools used for supporting the activities of software projects (e.g. requirements management, configuration management, coding environment, verification,

instruments,...) are tailored and suitable for the project characteristics may impact on the carbon footprint of the whole software development project. In fact, over-sized or inadequate tools make the project activities performance prone to waste of resources (as power or time).

- *Virtualization*: Virtualization allows accessing multiple applications on different operating systems at the same time, while abstracting (virtualizing) the machine hardware. The main focus of virtualization is to leverage the IT infrastructure (data center machines) in an optimal fashion (by making use of under utilized machines) to reduce operational costs. One of the effects of the virtualization is the possibility to optimize the resource usage as well as share resources that are dynamically acquired on-demand.
- *Estimation trust*: the accuracy of the estimations made at the beginning of the project determines the allocation of resources. Wrong estimations may cause over-usage of resources (e.g. people and tools) or the need to re-define the project resource allocation. In both the cases the impact on the carbon footprint of the project is negative.
- *Process maturity*: process maturity (or capability) is a complex concept often used with different meanings. Nevertheless, its core meaning is: the more the process maturity, the less the probability of missing the process objectives. Then the maturity of the processes directly or indirectly related to software development increases the confidence in a limited waste of resources.
- *Change management process effectiveness*: changes in projects are potential causes of re-estimations, re-planning, re-allocation of resources, or acquisition of new resources for the project with consequent risk of waste of resources. An effective management of changes in the project reduces that risk.
- *Avoidance of environmentally dangerous material*: the negative effects occur in the case environmentally dangerous materials are scattered in the environment.

Table 4. Exemplar Green-Drivers Table

Success Factor		Direct Effects				
		Power waste	Effort/Time waste	Material waste production	Paper waste	Fuel waste
People & Actions	Developers					
	Users					
	Top management					
	External agents					
	Project team		Project team members skill			Team member location
	Social interaction					
Project Content	Project characteristics	Project Activities Value		Avoidance of environmentally dangerous material		
	Project scope, goals, objectives					
	Resources	Development-supporting tools efficiency	Development-supporting tools suitability			
	Technology	Virtualization			Dematerialization	Teleconferences
Development processes	Requirements determination					
	Project management		Estimation trust			
	Use standard method		Process maturity			
	User participation					
	User training					
	Management of change		Change management process effectiveness			
Institution-al Context	Organizational properties					Flexi-work
	Environmental conditions					

- *Dematerialization*: the ability of producing and distributing project-related work products in electronic format instead of paper sheets (or other media), allows for saving paper and improving of the carbon footprint of the whole project.
- *Team member location*: the location of people involved in the project influences the travels, the working times, and the communication infrastructures of the projects. Consequently the location of team members may have a significant impact on the carbon footprint of the whole project.
- *Teleconferences*: having virtual meetings (i.e. meetings performed by means of teleconferences) can reduce dramatically the number of travels and consequently the carbon emissions due to them.
- *Flexi-work*: it is the ability to do work at a location other than the work office. The consequent saving is through reduced travel due to moving to office everyday. Additional benefits are also achieved in terms of traffic and infrastructural problems in the cities.

5.2 Phase 2: Sustainability Indicators derivation

Once Green Drivers have been identified, the proposed methodology asks for applying the GQM technique to derive Sustainability indicators.

In the following of this section the application of the GQM technique on some of the Green Drivers contained in Table 4 is described, with the aim of deriving sustainability indicators. Because of the exemplificative purpose of this section, only a sample of the Green Drivers in Table 4 is taken into account.

Example 1:

Green Driver: *Development Supporting Tools Suitability*

Goal: *Analyze the Development Supporting Tools Suitability for the purpose of improving with respect to the Effort/Time waste from the viewpoint of the project manager in the context of software development project*.

Question (from group Q2): *Is the whole set of functionalities/features of the tools used as support of the development used?*

Indicator: *Percentage of used functionalities/features of the tools supporting development actually used respect the full set of functionalities/features provided*

Example 2:

Green Driver: *Virtualization*

Goal: *Analyze the Virtualization for the purpose of improving with respect to the Power waste from the viewpoint of the project manager in the context of software development project*.

Question (from group Q3): *How many virtual servers are used for project-related activities?*

Indicator: *Percentage of used virtual servers vs. Percentage of used physical servers.*

Example 3:

Green Driver: *Dematerialization*

Goal: *Analyze the Dematerialization (of project's documents) for the purpose of improving with respect to the Paper waste from the viewpoint of the project manager in the context of software development project*.

Question (from group Q3): *How many documents are printed out by project team members?*

Indicator: *Number of hardcopies of project's documents.*

Example 4:

Green Driver: *Teleconference*

Goal: *Analyze the Teleconference (usage) for the purpose of improving with respect to the Fuel waste from the viewpoint of the project manager in the context of software development project*.

Question (from group Q1): *How many project meetings with stakeholders (e.g. suppliers, customer, other project team members located in different plants…) are performed by means of teleconference facilities?*

Indicator: *Percentage of teleconferences meeting vs. percentage of face-to-face meetings.*

6. CONCLUSIONS

In this paper we proposed a methodological approach to derive sustainability indicators to be used in the management of software development projects in terms of sustainability. The ultimate purpose is to give the project managers the capability to arrange activities and operate choices having the effect of reducing the carbon footprint of the whole project.

The presented methodology is composed of two main steps. The first one is aimed at identifying the Green Drivers (i.e. those characteristics of the software development project able to influence its environmental impact) by means of a bi-dimensional scheme that takes into account both the project success factors and the direct effects in the environment (i.e. those caused by the ICT infrastructures and equipment's in software development). Once the Green Drivers are derived, in order to express their impact in a quantitative and reusable way, as a second step, we of applied a tailored version of the GQM method to derive a set of measurable sustainability indicators for the project. To clarify the proposed methodology a set of exemplar Green Drivers and related indicators have been provided.

We consider this work an initial contribution because we are aware of the need of enforcing the proposed methodology. In particular, we intend to move forward according to the following principal directions:

- We intend to study the combined effects of different factors in the determination of the Green Drivers. In fact, it is possible that a Green Driver could have a positive influence on a Direct Effect but a negative one on another.
- We intend to make experiments on the application of the sustainability indicators in real software projects in order to get and share a feedback from the field on the validity of such an approach.
- As all aspects of any system/software lifecycle are determined by the related lifecycle processes [22, 23], we could investigate sustainability issues not just at project level, but also at organization processes level (which we think is the most appropriate level for sustainability), towards

the definition of a sustainability maturity model for software-intensive systems suppliers.

7. REFERENCES

[1] United Nations General Assembly. 1987. *Report of the World Commission on Environment and Development: Our Common Future*. Transmitted to the General Assembly as an Annex to document A/42/427 - Development and International Co-operation: Environment. 1987

[2] Sverdrup, H. and Svensson, M.G.E. 2005. Defining the Concept of Sustainability. In: Systems Approaches and Their Application. Springer, 143-164.

[3] World Economic Forum. 2009. *Green Technology: Driving Economical and Environmental Benefits from ICT*. Gordon J., Skinner A. and Button A. Editors.

[4] Hilty, L.M. and Lohmann, W. 2011. The Five Most Neglected Issues in "Green IT". CEPIS Upgrade *The European Journal for the Informatics Professional*. Vol. XII, issue No. 4, (October 2011). Council of European Professional Informatics Societies CEPIS Publisher, pp.11-15.

[5] Rivoire, S. 2007. Models and Metrics to Enable Energy-efficiency Optimization. *IEEE Computer*. (December 2007).

[6] Darnall, N. and Gallagher, D.R., Andrews, R.N.L. 2001. ISO 14001: Greening management systems. In *Greener Manufacturing and Operations*, SarkisJ (ed.). Greenleaf: Sheffield; 179–190.

[7] Miles, M.P. and Russel, G.P. 1997. ISO 14000 total quality environmental management: the integration of environmental marketing, total quality management, and corporate environmental policy. *Journal of Quality Management* 2 (1), 151–168.

[8] Capra E., and Merlo, F. 2009 Green IT: Everything Starts from the Software. *Proceedings of the 17th European Conference on Information Systems (ECIS 2009)*. Verona, Italy.

[9] Taina, J. 2011. Good, Bad, and Beautiful Software – In Search of Green Software Quality Factors. 2011. CEPIS Upgrade *The European Journal for the Informatics Professional*. Vol. XII, issue No. 4, October 2011. Council of European Professional Informatics Societies CEPIS Publisher, pp. 22-27.

[10] Lami G., Fabbrini F., and Fusani 2012. M. Software Sustainability from a Process-Centric Perspective, Proceedings of EuroSPI 2012, Vienna (Austria), June 2012. *Communication in Computer and Information Science CCIS* n. 301, D. Winkler, R.V. O'Conner, R. Messnarz (eds.), Springer 2012, pp. 97-108.

[11] Lami, G. and Buglione, L. 2012. Measuring Software Sustainability from a Process-Centric Perspective. *Proceedings of IWSM/MENSURA 2012*. Assisi, 17-19 October 2012. CPS publisher.

[12] Naumann, S., Dick, M., Kern, E. and Johann, T. 2001. The GREENSOFT Model: A Reference Model for Green and Sustainable Software and its Engineering. *Sustainable Computing: Informatics and Systems* 1 (2001), 294-304.

[13] Johann, T., Dick, M., Kern, E. and Naumann, S. 2011. Sustainable Development, Sustainable Software, and Sustainable Software Engineering. *Proceedings of the Int.l Symposium on Humanities, Science and Engineering Research*, Kuala Lampur 6-7 June 2011. IEEE Publisher.

[14] Tate, K. 2006. *Sustainable software development: An agile perspective*. Upper Saddle River, NJ: Addison-Wesley, 2006.

[15] Agarwal, S., Nath, A. and Chowdhury, D. 2012. Sustainable Approaches and Good Practices in Green Software Engineering: *International Journal of Research and Reviews in Computer Science*, Vol. 3, No. 1, February 2012.

[16] Chrissis, M.B., Konrad, M. and Shrum, S. 2004. *CMMI Guidelines for Process Integration and Product Improvement*. Addison-Wesley, 2004.

[17] International Organization for Standardization. 2011. ISO/IEC IS 15504, Information technology -- Process assessment, Parts 1-10, 2001-2011.

[18] Pamlin, D. and Szomolànyi, K. *First Report for Reduced CO2 emissions in the EU and Beyond*. Saving the Climate @ the Speed of Light joint ETNO and WWF initiative. 2008. DOI= http://www.etno.be/Portals/34/ETNO%20Documents/Sustainability/Climate%20Change%20Road%20Map.pdf.

[19] Murugesan, S. 2008. Harnessing Green IT: Principles and Practices. IEEE IT PRO (January/February 2008), 24-33.

[20] McLeod, L. and MacDonell, S.G. 2011. Factors that Affect Software Systems Development Project Outcomes: A Survey of Research. *ACM Computing Surveys*, Vol. 43, No. 4, Article 24, October 2011.

[21] Basili, V., Caldiera, G., and Rombach, D. 1994. Goal/question/metric paradigm. *Encyclopedia of Software Engineering*. Vol. 1, J. C. Marciniak, Ed. John Wiley and Sons, New York, 528-532.

[22] International Organization for Standardization. 2008. ISO/IEC 15288, Systems and software engineering — System life cycle processes.

[23] International Organization for Standardization. 2008. ISO/IEC 12207, Systems and software engineering — Software life cycle processes.