Christophe Ponsard[1], Raphael Michel[1],
Jean-Christophe Deprez[1], Sanae Saadaoui[1]

Guiding Cloud Application Developers towards a Balanced Design Trade-off among Energy Impacting Requirements[2]

Abstract

ICT energy efficiency is a growing concern. A great effort has already been put making hardware more energy efficient and aware. Although a part of that effort is devoted to specific software areas like embedded/mobile systems, much remains to be done at the software level, especially for applications deployed in the Cloud. There is an increasing need to help Cloud application developers to learn to reason about how much energy is consumed by their applications on the server side. This paper presents how to help developers to capture and deal with the interrelation between energy goals and other possibly conflicting non-functional requirements (NFRs) with the aim to guide them in the selection of a balanced compromise. We here extend our toolset, which already supports enabling energy awareness at requirements and design stages, with the ability to relate energy NFR with other kinds of NFRs. We also explore different design options based on collected measurements relating to those NFRs.

Keywords: Cloud Computing, energy efficiency, green IT, Goal-Oriented Requirements Engineering, Non-Functional Requirements, Data Visualization

1. Introduction

The expansion of ICT both at professional and personal levels induces the processing and exchange of increasingly larger amounts of data, increasing connectivity of all devices (mobile devices, Internet of Things) and higher penetration in all domains. This would raise the energy required to run ICT to a dramatic level if ICT energy efficiency was not improving simultaneously. However,

1 CETIC Research Centre, Gosselies, Belgium.
2 This work was partly funded by the European Commission under the FP7 AS-CETiC project (no. 610874). Thanks to the MegSUS reviewers for their useful comments.

because of this continuous increasing energy consumption[3] and in order to reach another level of energy saving, it is required to consider the software layer.

Several initiatives have already studied how to reduce energy consumption of mobile or embedded devices. For the Cloud Computing domain, an important amount of work has focused on lower layers such as the physical infrastructure[4] or on the infrastructure virtualisation layer[5]. A systematic survey of sustainability showed a dedicated attention to the Cloud as well as a number of proposals turned to the energy efficiency of software application[6]. However, much remains to be done, especially to help developers to learn how much energy is consumed by their application on the server side.

Unlike certain performance or security characteristics already understood by users and developers, energy consumption behaviour of server-side components is often largely unknown. Rare are those who could state quantifiable requirements on the energy consumption behaviour on the server side for particular features of their application.

In order to structure our work, we will refer to the previously defined reference framework[7]. This framework is composed of three levels:

1. **Requirements level** – We use the Goal-Question-Metric (GQM) paradigm[8]. In the previous work[9], we have shown how developers can formulate energy-related goals and questions in order to gain a more precise knowledge of the energy consumed by various features/components of their application. We also make the link with a number of already identified energy-related metrics[10].

---

3 InternetScience D8.1. Overview of ICT energy consumption, http://www.internet-science.eu (25/10/2015).

4 B. Dougherty, J. White, D.C. Schmidt, *Model-driven Auto-scaling of Green Cloud Computing Infrastructure*, "Future Generation Computer Systems" 2012, vol. 28, no. 2, pp. 371–378.

5 T. Mastelic et al., *Cloud Computing: Survey on Energy Efficiency*, "ACM Computing Surveys", Decembre 2014, vol. 47, no. 2, pp. 33:1–33:36.

6 B. Penzenstadler et al., *Systematic Mapping Study on Software Engineering for Sustainability (SE4S)*, 18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14, London, May 2014, pp. 14:1–14:14.

7 J.C. Deprez, C. Ponsard, *Energy Related Goals and Questions for Cloud Services*, "Measurement and Metrics for Green and Sustainable Software (MeGSuS)" 2014.

8 V.R. Basili, G. Caldiera, D.H. Rombach, *The Goal Question Metric Approach*, vol. I, John Wiley and Sons 1994.

9 J.C. Deprez, R. Ramdoyal, C. Ponsard, *Integrating Energy and Eco-Aware Requirements Engineering in the Development of Services-Based Applications on Virtual Clouds*, First International Workshop on Requirements Engineering for Sustainable Systems 2012.

10 P. Bozzelli, Q. Gu, P. Lago, *A Systematic Literature Review on Green Software Metrics*, Technical Report VU University, Amsterdam 2013.

2. **Design level** – To capture the information in a way that is both standard for an analyst and easy to process in further steps, a UML profile has been defined[11]. This profile enhances the analysis process of a Cloud application with energy awareness both for the development of a new application or the migration of an existing application to the Cloud. It also enables the automated deployment of measurement probes to monitor the specified Key Performance Indicators (KPI) and report them at the GQM level.
3. **Run-time level** – Probes collect the specified data and report them to a monitoring infrastructure part of the energy-aware Cloud stack. For this purpose, we currently use the ASCETiC Cloud stack deployed in specific test beds[12].

Energy requirements are part of the system NFRs, i.e. constraints on the way the software-to-be should satisfy its functional requirements or on the way it should be developed[13]. A number of taxonomies have been defined to classify NFRs, such as the NFR framework[14] or the SQuaRE standard[15].

In this paper, we aim at extending the previous work by not only trying to quantify the energy efficiency NFR, but also by considering it together with other NFRs that can potentially impact it in the context of a Cloud application. For example, performances like response time are typically conflicting because they require mobilizing enough resources. Security can also degrade energy efficiency because extra layers of software and communication overheads are necessary. In order to address this, we extend our previous work as follows:

1. **At the requirements level**, we capture other NFRs as goals as well as their relationship to energy goals. Using our method, other NFRs can also be refined to metrics that can be collected.
2. **At the design level**, our existing profile already supports the collection of extra metrics, our main contribution is made after the run-time data collection step. We provide a visualisation tool allowing developers to explore different design alternatives and guide them in the selection of a compromise.

---

[11] C. Ponsard, J.C. Deprez, J. Flamand, *A UML KPI Profile for Energy Aware Design and Monitoring of Cloud Services*, 10th International Conference on Software Engineering and Applications (ICSOFT-EA), July 2015.

[12] ASCETIC FP7 Project, http://www.ascetic.eu (25/10/2015).

[13] A. van Lamsweerde, *Requirements Engineering: from System Goals to UML Models to Software Specifications*, Wiley 2009.

[14] L. Chung, B.A. Nixon, E. Yu, J. Mylopoulos, *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publishers 2000.

[15] ISO/IEC: 25010:2011, *Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Quality Models*, ISO 2011.

3. **At the run-time level**, we support the collection of a wider set of metrics based on an extended set of probes, e.g. to collect response time related to performance. This part will not be detailed further here.

Our paper is organized as follows: Section 2 will present the structuring of energy related NFRs. Section 3 will focus on the guidance in exploring the space of energy-NFR compromise. Section 4 discusses some related work. Finally, section 5 draws conclusions and discusses our future work.

2. Capturing and Structuring Energy-related NFRs

This paper will not present an exhaustive catalogue of such NFRs but rather highlight our general design approach with a focus on Cloud computing. Given that our previous approach was goal-based, it is quite natural to consider a goal-oriented requirements engineering (GORE) framework to deal with such NFRs from a larger perspective, since it provides all the tools to capture NFR as well as their inter-relationship like contributions or conflicts. In the scope of this work we considered GRL and the related Open Source jUCMnav tool[16].

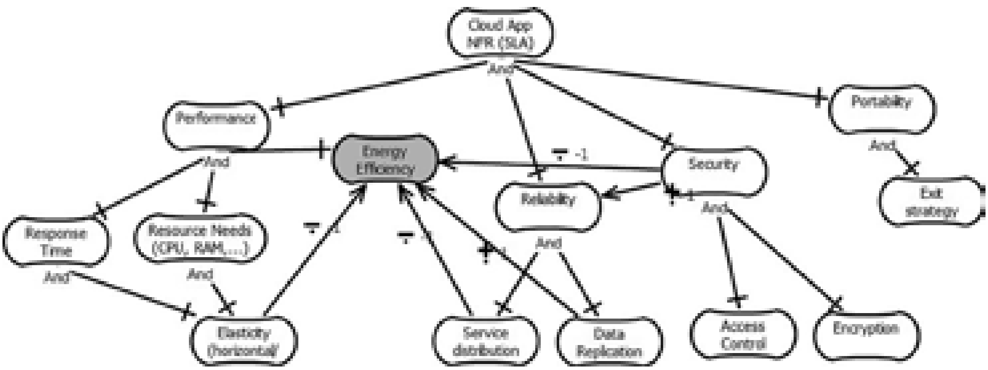

Figure 1. High Level Structure of NFR and Some Relationships

Source: the authors' self-study.

In order to identify the key NFRs, we also need to consider the concerned stakeholders, because the subsequent reasoning can be done with different sets

[16] jUCMNav: Juice up your modelling, http://jucmnav.softwareengineering.ca/ucm/bin/view/ProjetSEG (25/10/2015).

of stakeholders, depending on the kind of deployment considered. From a service customer's point of view, response time and price are the most common metrics used to evaluate services and applications, for these are the ones that will directly impact end users. From a service host's point of view, the overall cost is the most obvious parameter to minimize. This cost minimization is subject to multiple constraints which are affected by NFRs. For the service to remain useful and profitable, the quality of service must reach a certain level. Service Level Agreements (SLAs) contractually ensure that the required quality of service is achieved consistently by defining measurements.

We rely here on the large amount of work already carried out by several European projects[17]. Figure 1 structures the most frequently encountered NRFs such as availability, performance (response time, resource and also energy), security/privacy of data, location/access/portability of data, exit strategy using a structure similar to SQuaRE. A distinctive feature of our work is obviously the inclusion of energy NFRs, which also relates to "Green SLAs" that are being considered in the efficient use of resources, particularly energy, by services and applications[18]. At this high-level, we can state some generic conflicts, e.g. redundancy will increase energy consumption, security will also require more resources (CPU, transfer volume) to cope with encryption for example, and thus increase energy demand. However, other contributions might depend on the application, e.g. a good data replication strategy which can have a positive energy balance for data intensive applications. In the end, the assessment will need to be evaluated in the scope of a specific application which can be designed or deployed in different ways. This will require exploring a design space with the energy efficiency becoming a parameter of the total cost function.

Figure 1 was elaborated for a specific case study. The SQUaRE taxonomy was used as a check-list and all (sub) characteristics where not relevant here. However, our belief is that some of the identified relations can be generalized in order to be reused, for example under the form of patterns which could make the link with functional requirements (e.g. requirements on availability for data storage).

---

17 European Commission, *Cloud Computing Service Level Agreements – Exploitation of Research Results*, http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?doc_id=2496 – (25/10/2015).

18 M.E. Haque et al., *Providing Green SLAs in High Performance Computing Clouds*, International Green Computing Conference, IGCC, Arlington, VA, USA, June 2013.

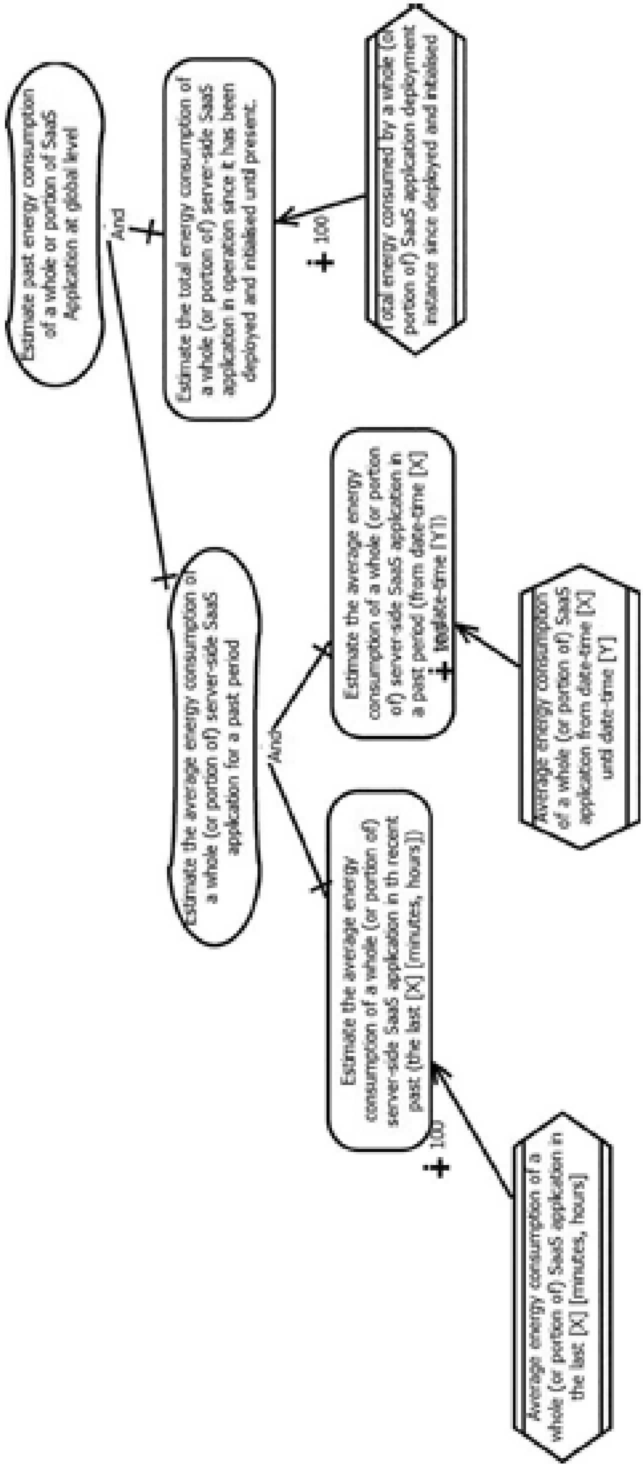

Figure 2. Estimation of Past Energy Consumption of a SaaS Application

Source: the authors' self-study.

Our approach is to keep the application designer in control of the design space exploration by (1) providing him or her with a specific visualization tool, which will be detailed in the next section, and (2) providing decision support taking the form of a measurable goal graph. This means that requirements need to have an associate KPI (identified from our KPI profile[19]) and that the refinement/contribution structure is decorated with quantitative information that can be computed by the tool. Such a structure is partly depicted in Figure 2, where the high-level energy-related goal is refined until a measurable level is reached (here using a temporal pattern).

3. Guiding the Developer in the Compromise Space

Finding the right trade-off between multiple concerns may quickly become a daunting problem as the complexity increases dramatically with each new parameter. To help service and application designers to find the right balance between their different concerns, we have developed a visualization tool that allows

1) comparing how the different parameters behave on different versions of an application or service, and
2) easily defining the trade-off and seeing which versions of the application or service match the constraints set by the trade-off. The tool is intended to be part of the ASCETiC Open Source toolbox.

3.1 Comparing Versions

The first and simplest visualization uses a simple chart to compare a given metric between multiple measurement sets. A measurement set is a set of measures taken in given conditions. For example, a measurement set may contain data about the response time, CPU, RAM, and energy usage of one virtual machine (VM) during one particular test (a search in a product catalogue for example), while another measurement set may contain the same information during the same test, but with a different version of the search code, or with a different application deployment. These tests are repeated multiple times and the measures are aggregated to get reasonable estimates.

[19] C. Ponsard et al., op.cit.

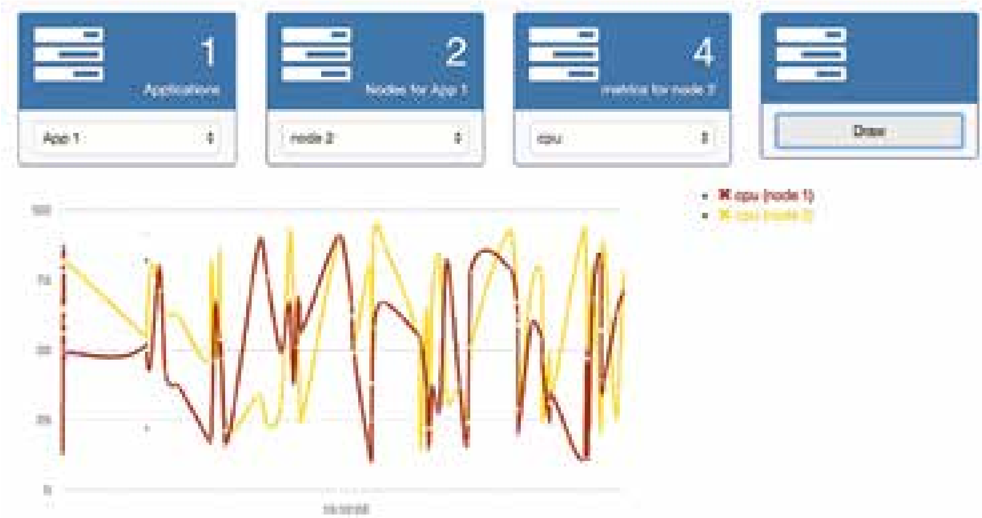

Figure 3. CPU Comparison between Two Nodes in the Same Test Run
Source: the authors' self-study.

Then, these measures can be compared in a graph. For example, the response time can easily be displayed using a bar chart, one bar for each measurement set, and the CPU usage will be displayed using a line chart, where the CPU usage can be visualized on the duration of the test. Figure 3 illustrates a visualization comparing CPU history on two nodes of the same experiment. This kind of graphs allows developers to:

- identify potential peaks or long CPU intensive operations that could be investigated further and optimized locally,
- compare two versions or two deployments of their code to see the actual effects of a change.

This visualization is focused on comparing one specific aspect between multiple versions of a service or application. The second visualization takes another point of view and visualizes tests and versions according to several of their characteristics.

3.2 The Trade-off Definition

The second view approaches the data set from the opposite side. Instead of taking different tests and versions and showing how they compare on one specific aspect, this view takes multiple aspects and for each of them, displays how the various measurement sets are distributed, then it allows filtering out the unwanted parts of the distributions and seeing which measurements sets match the filter.

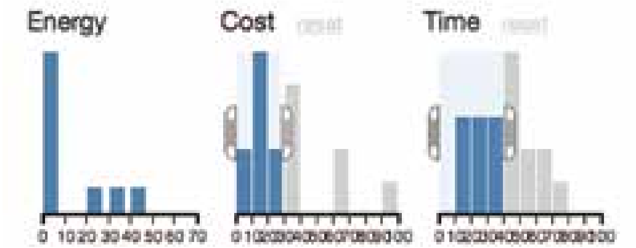

Figure 4. Filtering on Aspects

Source: the authors' self-study.

Figure 4 illustrates this concept more clearly: three bar graphs display the distributions of the measurements sets over the Energy consumption, Cost, and Time dimensions. In each chart, the X axis represents an arbitrary measure of the aspect, and the Y axis represents the corresponding number of measurements sets. We selected those three dimensions as the most relevant in our current experiments but other dimensions such as risk and continuity management can be explored, too. Our visualization is also not restricted to three dimensions.

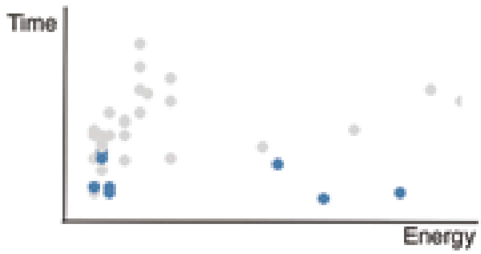

Figure 5. Energy (kWh) vs Time (s)

Source: the authors' self-study.

Figure 4 also shows the filtering mechanism. Using sliders in each graph, the user can easily select the parts of each chart that he or she wants to include in the filter. In this example, there is no restriction on the energy measures, there is a filter from 0 to 30 on the cost and from 0 to 40 on the time. The user can immediately see the impact of the filters on how the results are distributed on a scatter plot with two of the parameters. Figure 5 shows the result of the filtering on cost and time on the energy versus time plot. We can clearly see that the time dimension (the Y axis) has been capped at a given level: all the blue points are under the level selected by the filter. We can also see, however, that not all the points under that line are blue. On the left side for example, a few dots are grey, even though there is no filter on the energy (the X axis) aspect. This is an effect of the filter on the cost. The grey dots on the left side of the graph (low energy) had a higher cost, and they were excluded by the filter on the cost aspect.

The list of corresponding measurements sets and versions of the service that match the filters is also updated, so that the user can then make an informed choice about which version of his or her service he or she can ship and deploy.

4. Related Work and Discussion

The SQUaRE/ISO25010[20] (replacing ISO9126) provides a generic framework to systematically capture different application qualities. We rely on it to guide the high level structure of our NFRs decomposition. However, it is domain independent and also does not enable to capture relations between qualities, as it can be done by specific GORE frameworks like Chung's NFRs[21].

Overall, the SaaS KPI and visualization tools propose an operational approach to perform a particular assessment of the architecture trade-off analysis method (ATAM)[22] focused on runtime quality characteristics of various deployment alternatives of a SaaS application to be operated in Cloud infrastructures. However, unlike the traditional assessment with ATAM or other approaches to quantify NFRs, our approach assumes an executable application, e.g. a legacy application being migrated to operate in the Cloud or a SaaS application being developed incrementally. As both scenarios are commonplace, we believe that we will cover a significant percentage of Cloud application development projects.

On the visualization side, there exist other tools like the ClaferMoo Visualizer, a tool aimed at visualizing Pareto fronts to help making a choice between optimal configurations of a product line[23]. This work uses a 4 dimensional bubble chart by using the size and the colour of the bubble as additional visual

[20] ISO/IEC: 25010:2011, op.cit.

[21] L. Chung et al., op.cit.

[22] R. Kazman, M. Klein, P. Clements, *ATAM: Method for Architecture Evaluation*, Technical Report CMU/SEI-2000-TR-004, SEI, Carnegie Mellon University 2000.

[23] A. Murashkin, *Web-based GUI for Pareto Front Visualization and Analysis*, no. GSD-LAB-TR 2013–02–04, Waterloo, University of Waterloo 2013.

variables. Our work, while similar in appearance, takes a different approach. Our constraints can take the form of boundaries on different dimensions and we limit to use 2-dimensional charts to keep the comparison simple. Also, we do not address the issue of optimization.

This work can be generalized to some extent to other kinds of software applications as the GQM analysis and UML profile are generic, as well as the visualization tool. However, the elaborated models are specific to the Cloud domain. This domain knowledge is part of the tool delivered to the Cloud application developer. The global approach can certainly be applied to other domains but will only be worth the investment if it exhibits enough specificities/constraints and has a large enough developer base (e.g. mobile ecosystems, smart card product families, etc).

5. Conclusion and Future Work

In this paper, we presented an approach to help Cloud developers to make the right trade-offs among different energy related non-functional requirements. Our approach consists of a design phase where relationships and impacts among NFR are identified, followed by a run-time phase where the collected data are visualized for a set of design/deployment alternatives and an appropriate trade-off is identified inside the explored design space. This approach has been prototyped using Open Source technologies.

Our experimentation so far is still limited to partial data sets, which have been manually collected from a news publication application currently being migrated to the Cloud, and complemented by a test data generator. The next steps in our work will be to achieve a complete integration with the ASCETiC testbed and validate the approach in two other case studies (a microservice-based shopping cart and numeric simulation for building optimization). In this process, we also enrich our goal refinement pattern database to document links between NFR and energy effectiveness. In order to provide the best assistance to the Cloud application developer, we also plan to further develop the visualization capabilities of the tool and to integrate some data analysis capacities helping in the identification of interesting trade-offs.

References

ASCETIC: Adapting Service LifeCycle towards Efficient Clouds FP7 Project, http://www.ascetic.eu (25/10/2015).

Basili V.R., Caldiera G., Rombach D.H., *The Goal Question Metric Approach*, vol. I, John Wiley and Sons 1994.

Bozzelli P., Gu Q., Lago P., *A Systematic Literature Review on Green Software Metrics*, Technical Report, VU University, Amsterdam 2013.

Chung L., Nixon B.A., Yu E., Mylopoulos J., *Non-Functional Requirements in Software Engineering*, Kluwer Academic Publishers 2000.

Deprez J.C., Ponsard C., *Energy Related Goals and Questions for Cloud Services*, "Measurement and Metrics for Green and Sustainable Software (MeGSuS)" 2014.

Deprez J.C., Ramdoyal R., Ponsard C., *Integrating Energy and Eco-Aware Requirements Engineering in the Development of Services-Based Applications on Virtual Clouds*, First International Workshop on Requirements Engineering for Sustainable Systems 2012.

Dougherty B., White J., Schmidt D.C., *Model-driven Auto-scaling of Green Cloud Computing Infrastructure*, "Future Generation Computer Systems" 2012, vol. 28, no. 2, pp. 371–378.

European Commission, *Cloud Computing Service Level Agreements – Exploitation of Research Results*, http://ec.europa.eu/information_society/newsroom/cf/dae/document.cfm?doc_id=2496 (25/10/2015).

Haque M.E., et al., *Providing Green SLAs in High Performance Computing Clouds*, International Green Computing Conference, IGCC, Arlington, VA, USA, June 2013.

InternetScience D8.1. Overview of ICT energy consumption, http://www.internet-science.eu (25/10/2015).

ISO/IEC: 25010:2011, *Systems and Software Engineering – Systems and Software Quality Requirements and Evaluation (SQuaRE) – Quality Models*, ISO 2011.

jUCMNav: Juice up your modelling, http://jucmnav.softwareengineering.ca/ucm/bin/view/ProjetSEG (25/10/2015).

Kazman R., Klein M., Clements P., *ATAM: Method for Architecture Evaluation*, Technical Report, CMU/SEI-2000-TR-004, SEI, Carnegie Mellon University 2000.

Mastelic T., et al., *Cloud Computing: Survey on Energy Efficiency*, "ACM Computing Surveys", Decembre 2014, vol. 47, no. 2, pp. 33:1–33:36.

Murashkin A., *Web-based GUI for Pareto front Visualization and Analysis*, no. GSD-LAB-TR 2013–02–04, University of Waterloo, Waterloo 2013.

Penzenstadler B., et al., *Systematic Mapping Study on Software Engineering for Sustainability (SE4S)*, 18th International Conference on Evaluation and Assessment in Software Engineering, EASE '14, London, UK, May 2014, pp. 14:1–14:14.

Ponsard C., Deprez J.C., Flamand J., *A UML KPI Profile for Energy Aware Design and Monitoring of Cloud Services*, 10th International Conference on Software Engineering and Applications (ICSOFT-EA), July 2015.

van Lamsweerde A., *Requirements Engineering: from System Goals to UML Models to Software Specifications*, Wiley 2009.