A systematic literature review on Energy Efficiency in Cloud Software Architectures

A Systematic Literature Review on Energy Efficiency in Cloud Software Architectures

Giuseppe Procaccianti[a,b,*], Patricia Lago[a], Stefano Bevini[a]

[a] *VU University Amsterdam, De Boelelaan 1081a, Amsterdam, The Netherlands*
[b] *Politecnico di Torino, Corso Duca degli Abruzzi, 24, Torino, Italy*

Abstract

Cloud-based software architectures introduce more complexity and require new competences for migration, maintenance, and evolution. Although cloud computing is often considered as an energy–efficient technology, the implications of cloud–based software on energy efficiency lack scientific evidence. At the same time, energy efficiency is becoming a crucial requirement for cloud service provisioning, as energy costs significantly contribute to the Total Cost of Ownership (TCO) of a data center. In this paper, we present the results of a systematic literature review that investigates cloud software architectures addressing energy efficiency as a primary concern. The aim is to provide an analysis of the state-of-the-art in the field of energy-efficient software architectures.

Keywords: Cloud Computing, Energy Efficiency, Green Software, Software Architecture

1. Introduction

Information and Communication Technologies (ICT) energy demand is continuously increasing. Recent projections show that the fraction of commercial electricity consumed by ICT will be 10% of the total commercial electricity in the U.S. and almost 20% in Germany. In particular, projections for data centers in the U.S. indicate a growth in demand from 60 TWh/y in 2005 to 250 TWh/y in 2017 [1]. These figures show the need for more sustainable and energy efficient ICT technologies. Cloud computing is often regarded as to be one of those[2]. Indeed, one of the principles of cloud computing is on-demand provisioning of virtual resources, which can be aggregated on fewer physical machines. This allows to improve hardware utilization, thus increase energy efficiency.

Nowadays, energy efficiency is starting to be considered as a Service-Level Objective (SLO), i.e. a specific, measurable characteristic of a service, to be described as achievement values in Service-Level Agreement (SLA)[1]. An example would be: "The energy bill of the client should be reduced by 20% in one year". Cloud service providers could benefit from representing energy efficiency as a SLO.

However, in order to offer cloud services, providers rely on very complex software architectures. The impact of architecture characteristics on energy efficiency is yet unclear and possibly unexplored. Also, we still miss explicit or implicit reference architectures that can help in increasing energy efficiency.

The role of software in energy consumption is widely discussed among the scientific community, and a number of metrics for software energy efficiency have been proposed [3]. Our work tries to advance to the next step: whether it is possible to quantify the effects on energy consumption when adopting a certain software architecture, and what architectural solutions can be adopted to increase energy efficiency in cloud-based software.We performed a Systematic Literature Review (SLR) [4] to investigate the relationship between cloud-based software architectures and energy efficiency.

The preliminary results of our SLO were reported on an initial publication [5]. In this paper we extend our initial work, as follows: Section 2 describes our review protocol in detail. Section 3 presents the results of a demographic analysis conducted on our primary studies. Section 4 provides insights about the state–of–the–art of energy efficiency in cloud software architectures. Section 5 gives an overview of the stakeholders for energy efficiency we identified during our research. In Section 6 we discuss the threats to validity that might affect our study. Section 7 concludes the paper with future outlooks and follow-up studies.

2. Review Protocol

Based on the motivation introduced in Section 1, we identified the following Research Question (RQ) driving our study:

RQ. *What software architectural solutions for cloud service provisioning can be adopted to achieve Service Level Objectives on energy efficiency?*

In order to answer our RQ, we followed a systematic literature review process. We performed a preliminary analysis of the research space, and we identified 306 hits (i.e. potentially related studies). We formulated a review protocol for our study, by defining a search query for academic databases and inclusion and exclusion criteria. Applying the protocol, we selected the

*Corresponding Author.
Phone Number: +31 (0)20 598 7788, +31 (0) 644 549 209
Email addresses: g.procaccianti@vu.nl (Giuseppe Procaccianti), p.lago@vu.nl (Patricia Lago), stefano.bevini@gmail.com (Stefano Bevini)

[1] http://www.greenbiz.com/news/2009/01/12/energy-efficiency-new-sla, last visited on June 12th, 2013

primary studies for our research. We subsequently classified and analyzed these studies in order to extract relevant results.

In this section, we extensively describe our protocol, for the sake of reproducibility. All the main components of the protocol will be discussed: search strategy, study selection, data extraction, data analysis and traceability.

2.1. Search Strategy

We adopted *Google Scholar*[2] as our data source. We defined a query string by selecting the most appropriate keywords to answer our RQ. We selected five keywords: "software architecture", "cloud", "service", "SLO", "energy". Our query was defined after different steps, using the results of our preliminary analysis as pilot to test the coverage of the results. Namely, if one of the studies in our pilot was not retrieved by the query string, we refined it to add more keywords (typically, acronyms or alternative spellings, e.g. *"service level agreement"* vs. *"SLA"*.

The final query string was defined as follows:

"software architecture" AND cloud AND service AND "(energy OR power) efficiency" AND (SLA OR SLO OR "service level")

The query string was applied to titles, abstract and body of the studies, to enlarge the scope as much as possible. The search was conducted in June 2013, with a specified time range from 2000 to 2013.

2.2. Study Selection

In order to select our primary studies, we defined a number of criteria for inclusion and exclusion, (see Table B.3 in Appendix B). The criteria select papers in terms of their relevance to our RQ, but also in terms of scientific validity and language. In general, a study is selected if it fulfills all of the inclusion criteria, and excluded if it fulfills any of the exclusion criteria.

2.3. Data Extraction

We used an extraction form in order to retrieve and store relevant information about each primary study. Besides general information, the form records how energy efficiency is addressed and which architectural elements were identified in the presented solution. The extraction form is structured as follows:

- **Study Identifier**: provides an identifier for the study;
- **Study Title**: the publication title;
- **Study Type**: the publication type (i.e. journal article, conference article, thesis);
- **How energy efficiency is addressed**: a brief summary of how the presented solution addresses the energy efficiency of the cloud infrastructure;
- **Main architectural elements**: the main software elements of the solution.
- **Stakeholders**: stakeholders mentioned in the study that can be affected or involved in the architectural solution presented.
- **Validation**: whether the proposed solution has been validated in an Academic or Industrial setting, or no validation was performed. The validation is considered Academic when the article has been validated through a simulation or a test-bed. The validation is considered Industrial when the article reports a real case study (i.e. the proposed solution is already implemented in a software product).

2.4. Data Analysis

Our RQ investigates how cloud software architectures deal with energy efficiency issues. The aim of an SLR is to "identify, analyze and interpret all available evidence related to a specific research question" [6]. Hence, we do not aim at directly providing new reusable solutions or patterns, but rather we aim at classifying the existing body of knowledge in a systematic way.

To elicit this information, we adopted *coding*. Coding is a qualitative research method, commonly used in social sciences, that interprets data and organizes it in categories or families, using *codes*, i.e. words or short phrases. Coding allows to capture the fundamental information of qualitative data in a systematic way, and enables us to link it and discover patterns and trends [7]. Our first step was an exploratory study of the selected contributions, in order to define an initial set of codes (or "start-list" [8]). The start-list was built by analyzing reference literature in software architecture [9][10][11][12]. We then arranged our codes in a conceptual three-level structure, shown in Figure 1 and defined as follows:

- *Strategy*: the high-level approach through which a software solution addresses energy efficiency;
- *Technique*: the instantiation, or enactment, of a strategy through a specific technical approach;
- *Component*: an individual architectural component that plays a defined role in the application of a technique.

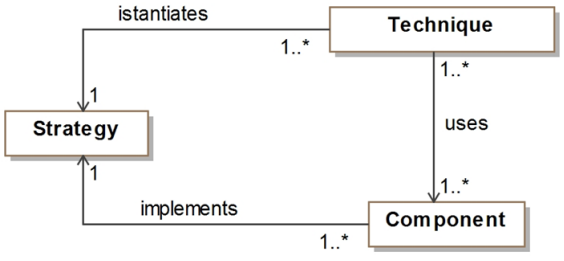

Figure 1: Conceptual structure of our codes.

The concept of architectural strategy [13], technique (or tactic) [9] and component [14] are very well known foundational

[2] http://scholar.google.com/

concepts of software architecture and they are familiar to practitioners and experts in the field. By adopting this conceptual structure, we aim at communicating our findings more effectively to software architects.

Finally, our primary studies were iteratively analyzed by two researchers independently, refining the list at every iteration until general and unambiguous codes were identified.

2.5. Traceability

We recorded the reference information of the studies using JabRef[3], a software tool for reference management. JabRef manages references in BibTeX and many other formats, and also allows to link and embed full-texts. For every step of the review process, a different JabRef database file was created that contained the references of the studies analyzed in that step. Moreover, we used an Excel sheet to report the matching of the inclusion/exclusion criteria and the stage of the decision (title, abstract or full-text checking) for each study. As regards the traceability of our analysis, whenever a code was identified in a primary study we annotated the corresponding section of the full-text of the contribution. In this way, the systematic mapping we performed can be verified by independent reviewers. All the material is available on request.

3. Demographic Analysis

In Table 1 we present the results of the application of our review protocol. The search query of Section 2 identified 149 initial results in the Google Scholar database. Then we went through the primary study selection process, divided into three phases: first, we checked the title against our inclusion/exclusion criteria, then we checked the abstract and finally the full-text. At each step, we were able to exclude a number of studies from our initial set and we finally ended up with 26 primary studies. The detailed list of our primary studies can be found in Table C.4 in Appendix C.

Step	*Removed*	*Remaining*
Initial search results	N/A	149
Title checking	10	139
Abstract checking	78	61
Full-text checking	35	**26**

Table 1: Overview of the selection process.

Figure 2 shows the distribution of our primary studies over time. It can be noted that the topic of energy efficiency in cloud service provisioning has become a concern in the past couple of years, as our primary studies have been mostly published starting from 2011. This distribution is coherent with the cloud computing hype cycle documented by Gartner [15]: in 2011 cloud computing was on top of the "Peak of Inflated Expectations". This reflects the growth of publications on cloud computing in 2011 that we can observe in Figure 2. No primary studies were identified in 2013: however, this is most likely due to the fact that when the search was performed (June 2013) many contributions published in the first months of the year were probably not indexed yet.

The distribution of the type of the articles is as follows:

- 11 journal articles;
- 12 conference articles;
- 3 PhD theses.

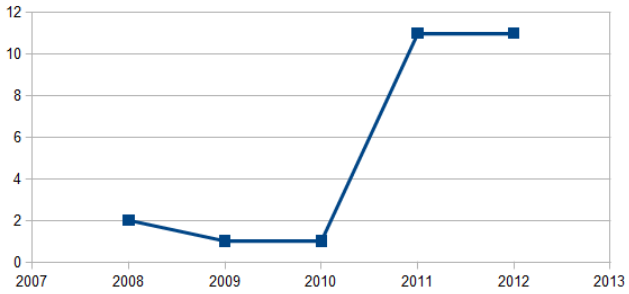

Figure 2: Number of primary studies selected per year.

As regards the validation of the solutions presented in the primary studies, we found that:

- 14 studies present an *Academic* validation;
- 2 studies present an *Industrial* validation;
- 10 studies do not validate the presented solution.

This testifies the low level of maturity of this topic. The lack of industrial validation indicates that the state-of-the-practice of energy efficiency in cloud software architectures has still to be determined.

4. Energy Efficiency in Software Architectures

Our results provide many insights on how energy efficiency is addressed by software architectures. As introduced in Section 2.4, results have been classified in terms of Strategies, Techniques and Components.

4.1. Strategies

We identified three strategies in the primary studies, namely:

- **Energy Monitoring**: this strategy is identified when some components of the software architecture of the presented solution are devoted at monitoring energy consumption;
- **Self Adaptation**: this strategy is identified when some components of the software architecture of the presented solution enable the possibility of adapting the software behavior in order to increase energy efficiency;

[3] `http://jabref.sourceforge.net/`, last visited on January 29th, 2014.

- **Cloud Federation**: this strategy is identified when the software architecture of the presented solution comprehends the possibility to "lease" or "negotiate" the usage of cloud services from other providers according to energy consumption requirements.

The following list enumerates the occurrences of the different strategies and their combinations among the articles.

- *Energy Monitoring*: identified in 1 primary study.
- *Self-Adaptation*: identified in 11 primary studies.
- *Cloud Federation*: identified in 3 primary studies.
- *Energy Monitoring* + *Self-Adaptation*: identified in 8 primary studies.
- *Energy Monitoring* + *Cloud Federation*: absent.
- *Self-Adaptation* + *Cloud Federation*: identified in 1 primary study.
- *Energy Monitoring* + *Self-Adaptation* + *Cloud Federation*: identified in 2 primary studies.

The overall adoption of the identified strategies is as follows:

- *Energy Monitoring* has been adopted, alone or in combination with other strategies, in 42% of our primary studies (11 out of 26)
- *Self-Adaptation* has been adopted, alone or in combination with other strategies, in 81% of our primary studies (21 out of 26)
- *Cloud Federation* has been adopted, alone or in combination with other strategies, in 23% of our primary studies (5 out of 26)

Among the three architectural strategies we identified, Self-Adaptation is the most adopted. Energy Monitoring is almost never adopted in isolation, but most of the time (i.e. 10 out of 11 studies) it is combined with Self-Adaptation. This suggests that Energy Monitoring techniques are usually adopted as enablers for Self-Adaptation techniques, providing necessary information to drive the adaptation process. The low adoption of Cloud Federation techniques might be due to the fact that multi-cloud environments are still uncommon, mostly due to standardization and security concerns [16, 17].

4.2. Techniques

For each strategy, we identified a number of techniques, through which the strategy is enacted. In Figure 3 we show the distribution of the techniques among the primary studies. A more detailed description can be found in Table D.6 in Appendix D. As for strategies, techniques are not applied in isolation: in all primary studies, more than one technique per study is applied.

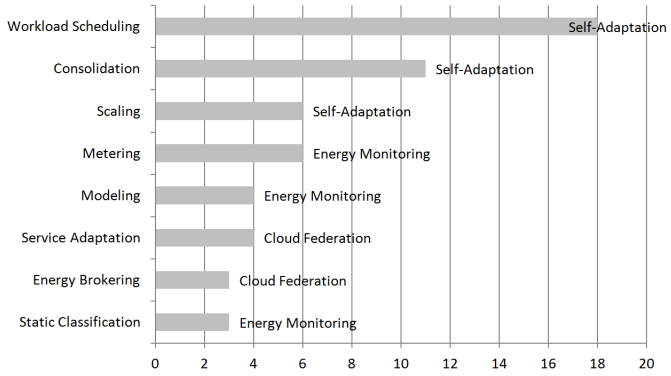

Figure 3: Distribution of techniques among the primary studies.

From Figure 3 a clear trend emerges: Consolidation and Workload Scheduling are by far the most adopted techniques (i.e. identified 11 and 18 times of 26 studies, respectively). Both of these techniques are already commonly used in cloud systems to improve their performance. This result is hence realistic and not surprising.

Another interesting finding is that many techniques exhibit dependencies between each other. For example, we observed that scheduling algorithms or Virtual Machine (VM) allocation processes are typically driven by components responsible for monitoring the infrastructure/system energy consumption. This implies that Workload Scheduling and Consolidation techniques depend on Metering and/or other Energy Monitoring techniques. Another dependency lies between the two Cloud Federation techniques: Service Adaptation needs an Energy Broker in order to retrieve the energy information of services and perform the service switching.

4.3. Components

For each strategy, we identified the software architecture components primarily responsible for its implementation. In Figure 4 we present their distribution among the primary studies. A more detailed description can be found in Table E.7 in Appendix E. The relationship between components and techniques is many–to–many: a technique uses a number of components and each component can be used in more than one technique.

The high frequency of Workload Scheduling and Consolidation techniques is also reflected in terms of components, as shown in Figure 4: as expected, the Workload Scheduler and the VM Allocator are the second and third most frequent component identified (i.e. 14 and 13 times out of 26 studies, respectively). There are cases in which the component is found outside of its most typical technique: that is because in those cases, the component plays a role that does not implement that technique. For example, in [18] the VM Allocator is not used in a Consolidation technique but rather in a particular case of a Scaling technique.

The high number of occurrences of the SLA Violation Checker is one of our key findings. In particular, they are present

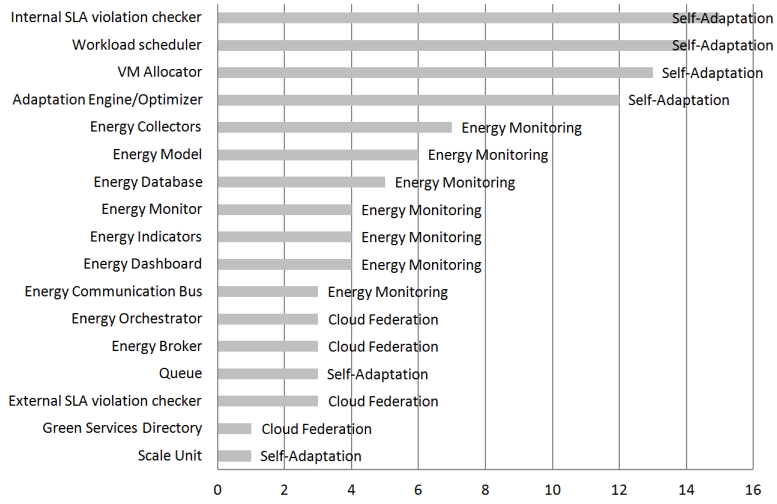

Figure 4: Distribution of components among the primary studies.

in both of the primary studies that received an industrial validation. This suggests that the trade-off between energy efficiency and other software quality aspects appears to be a major architectural concern. In particular, it is relevant to notice the difference between the Internal and External component: the Internal SLA Violation Checker monitors the fulfillment of the SLAs when performing Self-Adaptation techniques (i.e. Scaling, Consolidation or Workload Scheduling) so it typically has to pose constraints to the internal adaptation process. The External SLA Violation Checker instead enforces that when negotiating services between different providers, the resulting service composition matches the required quality of service for a certain task. That is, the External SLA Violation Checker poses constraints to the service composition process.

5. Stakeholder Overview

An important part of our data analysis focuses on the stakeholders that could have been affected or interested by the architectural solution introduced in the primary studies. Our aim is to identify stakeholders for energy efficiency, whose concerns can be targeted as an architectural concern. In Table 2 we show the stakeholders we identified, along with their definition and the criteria behind their identification. They are mentioned with the following frequency:

- *End-User*: mentioned in 6 primary studies.
- *Service Provider*: mentioned in 10 primary studies.
- *System Architect*: mentioned in 13 primary studies.
- *Infrastructure Manager*: mentioned in 12 primary studies.

From these numbers, we can observe that Infrastructure Managers and System Architects are, as expected, the most involved by the solutions identified in the primary studies. However, we also notice that End Users are the least involved. This implies that the End User is less aware of the benefits that the solution brings in terms of energy efficiency. Increasing user awareness could instead justify a trade-off between energy efficiency and other crucial quality attributes for the End User (such as performance or usability).

6. Threats to Validity

The evidence reported in our work is not immune to validity threats. With respect to the classification done by Wohlin et al. [45], we identify two types of threats, regarding **internal** and **external** validity.

As regards **internal** validity, the main threat concerns the effectiveness of our *search strategy*, as we chose to use only the Google Scholar search engine instead of multiple bibliographic databases. This choice was done after interviewing experts in the field of SLRs in SE. Google Scholar has substantially improved its coverage in the last few years [46] and it is now regarded as an appropriate and comprehensive source [47]. An additional bonus is that this choice simplified the implementation of our search, allowing us to focus more on data extraction and analysis.

Another concern to internal validity regards the *selection process* of the primary studies, as it was carried out by a single researcher. This might have introduced subjective bias in the process. To mitigate those risks, we carefully defined our inclusion/exclusion criteria, to make them as objective as possible. Moreover, the selection process was also carried out in multiple steps (title, abstract and full-text checking) to reduce misinterpretations to a minimum. We adopted a conservative approach, so we are more prone to Type I errors (i.e. false positives) rather than Type II (i.e. false negatives, exclusion of relevant studies).

The main threat to **external** validity is to be found in the data analysis phase. We adopted a *coding* technique to classify the architectural concepts extracted from the primary studies. As coding is a qualitative analysis method, it may be affected by interpretation bias of the individual researcher. To mitigate

Stakeholder	Definition	Identification Criteria	Occ.	References
End User	The actual user of the Cloud service.	The proposed architectural solution has a visible impact on the service presented to the end user.	6	[19] [20] [21] [22] [23] [24]
Service Provider	The provider of the Cloud service.	The proposed architectural solution explicitly monitors the SLAs fulfillment.	10	[25] [26] [27] [20] [28] [29] [30] [23] [24] [31]
System Architect	The main responsible of the system design [32].	The proposed architectural solution implies an intervention on the business logic of the software system (e.g., invasive monitoring, auto-scaling applications...)	13	[25] [33] [19] [34] [26] [20] [22] [35] [36] [23] [37] [38] [24]
Infrastructure Manager	The responsible for the optimal use of system resources.	The proposed architectural solution implies only an internal reorganization of the computing resources (e.g., consolidation)	12	[39] [18] [40] [41] [28] [42] [29] [30] [43] [44] [38] [31]

Table 2: Overview of the identified stakeholders for energy efficiency.

this risk, the coding process was performed independently by two different researchers, and the resulting lists of codes were merged upon discussion and agreement.

Most of our primary studies present solutions that were never validated in an industrial setting: some of them were validated in academic contexts, through simulation or other similar techniques, while others were not validated at all. Assessing the efficacy of these solutions in tackling energy efficiency issues is out of the scope of this SLR. Nevertheless, we have to consider the lack of validation as a threat to external validity, because it might affect the generalization of our findings. To mitigate this threat, we described the identified architectural concepts in a structured taxonomy, grounded in literature, along with a definition for each concept, hence reducing their specificity to a minimum. This will allow to apply these concepts in real-world case studies, where the impact of our findings on the energy efficiency of software architectures will be properly assessed.

7. Conclusions

As data centers are major power consumers, energy efficiency has become a primary issue for cloud service providers. In this context, both the hardware configuration and the software architecture of the cloud computing infrastructure must be carefully designed in order to accommodate power consumption constraints.

In this work, we performed a systematic literature review to research how energy efficiency is addressed by cloud software architectures. Our search resulted in 26 primary studies, mostly published in the last 3 years, each describing a software solution for energy efficiency. Through a coding process, we were able to structure these software solutions in terms of strategies, techniques and components. These concepts provide a common ground for architects to describe, analyze and design energy efficient software solutions.

We identified 3 main strategies: Energy Monitoring, Self-Adaptation and Cloud Federation. It emerged that Self-Adaptation is the most adopted strategy to achieve energy efficiency. However, Cloud Federation will need much more research in the future, due to the diffusion of multi-cloud environments and the need of optimizing the usage of Cloud infrastructures. Regardless of the adopted strategy, fulfilling SLAs constitutes a major concern for software architects. Trade-offs between energy efficiency and other quality attributes are to be further investigated, in order to predict the impact of energy efficient solutions on other service aspects.

We also investigated the stakeholders mentioned in our primary studies. Our results indicate that End-Users are the least involved, which also implies they are less aware of what software does to reduce its energy consumption. Given the massive scale of diffusion of software and services, even a small improvement could contribute greatly. Hence, increasing user awareness could lead to both environmental and economic benefits, as also pointed out by the European Commission in the Horizon 2020 Framework[4]. More research is needed to investigate what to communicate to the user, and how [48].

This work gives a systematic analysis of the state-of-the-art in energy-efficient cloud software architectures. In future research, we will extract reusable software solutions (i.e. tactics, design or architectural patterns) [49] for designing energy efficient software systems. In addition, as most of our primary studies were never validated in industrial settings, we plan to set up case studies to analyze real-world software architectures and how they deal with energy efficiency.

[4] http://ec.europa.eu/research/participants/portal/desktop/en/opportunities/h2020/topics/2360-ee-10-2014.html

Acknowledgment

This work has been partially sponsored by the European Fund for Regional Development under project MRA Cluster Green Software.

References

[1] B. Aebischer, L. M. Hilty, The energy demand of ICT: A historical perspective and current methodological challenges, in: ICT Innovations for Sustainability, Advances in Intelligent Systems and Computing, Springer International Publishing, 2015, pp. 71–103.

[2] A. Berl, E. Gelenbe, M. Di Girolamo, G. Giuliani, H. De Meer, M. Q. Dang, K. Pentikousis, Energy-efficient cloud computing, The Computer Journal 53 (2010) 1045–1051.

[3] P. Bozzelli, Q. Gu, P. Lago, A systematic literature review on green software metrics, Technical Report, VU University Amsterdam, 2013.

[4] B. Kitchenham, O. Pearl Brereton, D. Budgen, M. Turner, J. Bailey, S. Linkman, Systematic literature reviews in software engineering–a systematic literature review, Information and software technology 51 (2009) 7–15.

[5] G. Procaccianti, S. Bevini, P. Lago, Energy efficiency in cloud software architectures, in: Proceedings of the 27th Conference on Environmental Informatics - Informatics for Environmental Protection, Sustainable Development and Risk Management, volume 1, Shaker Verlag GmbH, 2013, pp. 291–299.

[6] B. A. Kitchenham, S. Charters, Guidelines for performing systematic literature reviews in software engineering (version 2.3), Technical Report EBSE-2007-01, EBSE, 2007.

[7] J. Saldana, The Coding Manual for Qualitative Researchers, English short title catalogue Eighteenth Century collection, SAGE Publications, 2012.

[8] M. B. Miles, A. M. Huberman, Qualitative data analysis: An expanded sourcebook, Sage, 1994.

[9] L. Bass, P. Clements, R. Kazman, Software architecture in practice, Addison-Wesley, 2012.

[10] ISO/IEC, Systems and Software Engineering–Architecture Description, Technical Report, ISO/IEC/IEEE 42010, 2011.

[11] D. E. Perry, A. L. Wolf, Foundations for the study of software architecture, ACM SIGSOFT Software Engineering Notes 17 (1992) 40–52.

[12] R. C. De Boer, R. Farenhorst, P. Lago, H. Van Vliet, V. Clerc, A. Jansen, Architectural knowledge: Getting to the core, in: Software Architectures, Components, and Applications, Springer, 2007, pp. 197–214.

[13] R. Kazman, J. Asundi, M. Klein, Quantifying the costs and benefits of architectural decisions, in: Software Engineering, 2001. ICSE 2001. Proceedings of the 23rd International Conference on, 2001, pp. 297–306.

[14] D. Garlan, M. Shaw, An introduction to software architecture, Advances in software engineering and knowledge engineering 1 (1993) 1–40.

[15] D. M. Smith, Hype cycle for cloud computing 2011, Gartner Inc., Stamford (2011).

[16] N. Ferry, A. Rossini, F. Chauvel, B. Morin, others, Towards model-driven provisioning, deployment, monitoring, and adaptation of multi-cloud systems, CLOUD 2013: IEEE (2013).

[17] A. Brogi, A. Ibrahim, J. Soldani, J. Carrasco, J. Cubo, E. Pimentel, F. D'Andria, SeaClouds: A european project on seamless management of multi-cloud applications, SIGSOFT Softw. Eng. Notes 39 (2014) 1–4.

[18] B. P. Dougherty, Configuration and Deployment Derivation Strategies for Distributed Real-time and Embedded Systems, Ph.D. thesis, Vanderbilt University, 2011.

[19] L. Curtis, Environmentally sustainable infrastructure design, The Architecture Journal 18 (2008) 2–8.

[20] T. Forell, D. Milojicic, V. Talwar, Cloud management: Challenges and opportunities, in: Parallel and Distributed Processing Workshops and Phd Forum (IPDPSW), 2011 IEEE International Symposium on, IEEE, 2011, pp. 881–889.

[21] S. K. Garg, C. S. Yeo, R. Buyya, Green cloud framework for improving carbon efficiency of clouds, in: Euro-Par 2011 Parallel Processing, Lecture Notes in Computer Science, Springer Berlin Heidelberg, 2011, pp. 491–502.

[22] G. Katsaros, J. Subirats, J. Oriol Fitó, J. Guitart, P. Gilet, D. Espling, A service framework for energy-aware monitoring and vm management in clouds, Future Generation Computer Systems InPress (2012) InPress.

[23] D. Villegas, N. Bobroff, I. Rodero, J. Delgado, Y. Liu, A. Devarakonda, L. Fong, S. Masoud Sadjadi, M. Parashar, Cloud federation in a layered service model, Journal of Computer and System Sciences 78 (2012) 1330–1344.

[24] L. Xu, G. Tan, X. Zhang, J. Zhou, Energy aware cloud application management in private cloud data center, in: Cloud and Service Computing (CSC), 2011 International Conference on, IEEE, 2011, pp. 274–279.

[25] A. Tzanakaki, M. Anastasopoulos, K. Georgakilas, J. Buysse, M. De Leenheer, C. Develder, S. Peng, R. Nejabati, E. Escalona, D. Simeonidou, N. Ciulli, G. Landi, M. Brogle, A. Manfredi, E. Lopez, J. F. Riera, J. A. Garcia-Espin, P. Donadio, G. Parladori, J. Jimenez, Energy efficiency in integrated IT and optical network infrastructures: The GEYSERS approach, in: Computer Communications Workshops (INFOCOM WKSHPS), 2011 IEEE Conference on, ieeexplore.ieee.org, 2011, pp. 343–348.

[26] F. G. A. De Oliveira Jr, T. Ledoux, et al., Self-optimisation of the energy footprint in service-oriented architectures, in: Proceedings of the 1st Workshop on Green Computing, 2010, pp. 4–9.

[27] C. Dupont, G. Giuliani, F. Hermenier, T. Schulze, A. Somov, An energy aware framework for virtual machine placement in cloud federated data centres, in: Future Energy Systems: Where Energy, Computing and Communication Meet (e-Energy), 2012 Third International Conference on, IEEE, 2012, pp. 1–10.

[28] N. Huber, F. Brosig, S. Kounev, Model-based self-adaptive resource allocation in virtualized environments, in: Proceedings of the 6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems, ACM, 2011, pp. 90–99.

[29] A. Noureddine, R. Rouvoy, L. Seinturier, Supporting energy-driven adaptations in distributed environments, in: Proceedings of the 1st Workshop on Middleware and Architectures for Autonomic and Sustainable Computing, ACM, 2011, pp. 13–18.

[30] A. Noureddine, R. Rouvoy, L. Seinturier, A review of middleware approaches for energy management in distributed environments, Software: Practice and Experience 00 (2012) 1–30.

[31] L. Xu, G. Tan, X. Zhang, J. Zhou, A bdi agent-based approach for cloud application autonomic management, in: Cloud Computing Technology and Science (CloudCom), 2012 IEEE 4th International Conference on, IEEE, 2012, pp. 574–577.

[32] J. A. Mills, A pragmatic view of the system architect, Commun. ACM 28 (1985) 708–717.

[33] J. Carpentier, J.-P. Gelas, L. Lefevre, M. Morel, O. Mornard, J.-P. Laisne, Compatibleone: Designing an energy efficient open source cloud broker, in: Cloud and Green Computing (CGC), 2012 Second International Conference on, IEEE, 2012, pp. 199–205.

[34] D. Rogers, U. Homann, Application patterns for green it, The Architecture Journal 18 (2008) 16–21.

[35] S. Gtz, C. Wilke, S. Cech, U. Assmann, Runtime variability management for energy-efficient software by contract negotiation, in: Proceedings of the 6th International Workshop Models@run.time (MRT 2011), 2011.

[36] S. Kounev, Self-aware software and systems engineering: A vision and research roadmap, GI Softwaretechnik-Trends 31 (4) (2011) 21–25.

[37] Z. Wu, M. E. Cotterell, S. Qin, A. Beach, G. Santiago, Towards a cloud infrastructure for energy informatics, in: Energy Informatics. Sprouts: Working Papers on Information Systems, 2012.

[38] N. Xiong, W. Han, A. Vandenberg, Green cloud computing schemes based on networks: a survey, Communications, IET 6 (2012) 3294–3300.

[39] P. J. Chacin Martnez, et al., A Middleware framework for self-adaptive large scale distributed services, Ph.D. thesis, Universitat Politecnica de Catalunya, Departament d'Arquitectura dels Computadors, 2011.

[40] M. Harman, K. Lakhotiaa, J. Singerb, D. R. Whiteb, S. Yooa, Cloud engineering is search based optimization too, Journal of Systems and Software 86 (2013) 2225–2241.

[41] M. Hedwig, Taming energy costs of large enterprise systems through adaptive provisioning, Ph.D. thesis, Albert-Ludwigs-Universitt Freiburg, Wirtschafts- Und Verhaltenswissenschaftliche Fakultt, 2009.

[42] L. Lu, P. J. Varman, K. Doshi, Decomposing workload bursts for efficient storage resource management, Parallel and Distributed Systems, IEEE Transactions on 22 (2011) 860–873.

[43] J. Sekhar, G. Jeba, S. Durga, A survey on energy efficient server consolidation through vm live migration, International Journal of Advances in Engineering & Technology 5 (1) (2012) 515–525.
[44] M. Sevalnev, S. Aalto, J. Kommeri, T. Niemi, Using queuing theory for controlling the number of computing servers, in: Proceedings of the 3rd International Conference on Green IT Solutions (ICGREEN 2012), SciTePress, 2012.
[45] C. Wohlin, Experimentation in Software Engineering: An Introduction, The Kluwer International Series in Software Engineering, Kluwer Academic, 2000.
[46] P. Younger, Using google scholar to conduct a literature search, Nursing Standard 24 (2010) 40–46.
[47] N. Bakkalbasi, K. Bauer, J. Glover, L. Wang, Three options for citation tracking: Google scholar, scopus and web of science, Biomedical digital libraries 3 (2006) 7.
[48] P. Lago, N. Meyer, M. Morisio, H. A. Müller, G. Scanniello, Leveraging "Energy Efficiency to Software Users": summary of the Second GREENS workshop, at ICSE 2013, SIGSOFT Softw. Eng. Notes 39 (2014) 36–38. URL: http://doi.acm.org/10.1145/2557833.2557859. doi:10.1145/2557833.2557859.
[49] G. Procaccianti, P. Lago, G. Lewis, Green architectural tactics for the cloud, in: Proceedings of the 11th Working IEEE/IFIP Conference on Software Architecture (WICSA), 2014. To appear.

Appendix A. Acronym List

TCO Total Cost of Ownership

ICT Information and Communication Technologies

SLO Service-Level Objective

SLA Service-Level Agreement

SLR Systematic Literature Review

RQ Research Question

VM Virtual Machine

Appendix B. Inclusion–Exclusion Criteria

Table B.3 summarizes the Inclusion–Exclusion Criteria for our review protocol.

Appendix C. List of Primary Studies

Tables C.4, C.5 present the list of the primary studies we identified during our SLR.

Appendix D. List of Techniques

Table D.6 presents the list of the primary studies we identified during our SLR.

Appendix E. List of Components

Table E.7 presents the list of the primary studies we identified during our SLR.

Criterion	*Rationale*
I1 *A study that directly proposes software architectures, architectural styles or strategies, or indirectly proposes them from a service provisioning perspective.*	We want to identify how software architectures affect energy efficiency, thus we need articles proposing software architectures, or indirectly proposing them from a service provisioning perspective.
I2 *A study that addresses energy efficiency as a quality attribute.*	We want to investigate whether energy efficiency is considered, by providers or experts, as a quality attribute for cloud services.
I3 *A study that is developed by either of academics and practitioners.*	Both academic and industrial solutions are relevant to this study.
I4 *A study that is published in software engineering/cloud computing field.*	Software engineering is our reference field, but cloud computing research can provide us an insight on what trends are set in terms of software architectures for cloud.
I5 *A study that is peer-reviewed.*	A peer-reviewed paper guarantees a certain level of quality and contains reasonable amount of content.
I6 *A study that is written in English.*	For feasibility reasons papers written in other languages than English are excluded.
E1 *A study that does not propose software solutions for energy efficiency.*	Traditionally, energy efficiency has been regarded as an hardware issue. We want to drive past this assumption and address the software impact of power consumption.
E2 *A study that does not imply any type of service provisioning.*	We are not interested in solutions that generally increase the energy efficiency of a datacenter, without having in mind how to provide an energy-efficient service to a customer.
E3 *A study that does not consider energy efficiency as a primary quality attribute.*	We are not interested in studies that consider energy efficiency a secondary concern.
E4 *A study that does not aim at optimizing the energy efficiency of the cloud computing infrastructure.*	Mobile devices often leverage cloud services by offloading computation tasks, in order to increase their battery life. Although this is an energy efficiency improvement, it is not relevant for the energy efficiency of the cloud computing infrastructure, thus we want to exclude these solutions from our study.

Table B.3: Inclusion and Exclusion Criteria.

Title	Authors	Year	Publication Type	Venue	Validation	Reference
Application patterns for green IT	Rogers, D. and Homann, U.	2008	Journal	The Architecture Journal (MSDN)		[34]
Environmentally Sustainable Infrastructure Design	Curtis, L.	2008	Journal	The Architecture Journal (MSDN)		[19]
Taming energy costs of large enterprise systems through adaptive provisioning	Hedwig, M.	2009	Ph.D. Thesis		Academic	[41]
Self-optimization of the energy footprint in Service-Oriented Architectures	De Oliveira, J. et al.	2010	Conference	1st International Workshop on Green Computing Middleware (GCM'2010)	Academic	[26]
A Middleware framework for self-adaptive large scale distributed services	Chacin Martinez, P. J. et al.	2011	Ph.D. Thesis		Academic	[39]
Cloud management: Challenges and opportunities	Forell, T. et al.	2011	Conference	2011 IEEE International Parallel & Distributed Processing Symposium	Industrial	[20]
Configuration and Deployment Derivation Strategies for Distributed Real-time and Embedded Systems	Dougherty, B.P.	2011	Ph.D. Thesis		Academic	[18]
Decomposing Workload Bursts for Efficient Storage Resource Management	Lu, L. et al.	2011	Journal	IEEE Transactions on Parallel and Distributed Systems	Academic	[42]
Energy aware cloud application management in private cloud data center	Xu, L. et al.	2011	Conference	International Conference on Cloud and Service Computing (CSC)	Industrial	[24]
Energy Efficiency in integrated IT and optical network infrastructures: The GEYSERS approach	Tzanakaki et al.	2011	Conference	IEEE Conference on Computer Communications Workshops		[25]
Green Cloud Framework for Improving Carbon Efficiency of Clouds	Garg, S. K. et al.	2011	Conference	17th International Conference on Parallel Computing (EURO-PAR 2011)		[21]
Model-based self-adaptive resource allocation in virtualized environments	Huber, N. et al.	2011	Conference	6th International Symposium on Software Engineering for Adaptive and Self-Managing Systems (SEAMS 2011)	Academic	[28]
Runtime Variability Management for Energy-efficient Software by Contract Negotiation	Götz, S. et al.	2011	Conference	Proceedings of the 6th International Workshop Models@run.time (MRT 2011)		[35]
Self-Aware Software and Systems Engineering: A Vision and Research Roadmap	Kounev, S.	2011	Journal	GI Softwaretechnik-Trends	Academic	[36]
Supporting energy-driven adaptations in distributed environments	Noureddine, A. et al.	2011	Conference	Proceedings of the 1st Workshop on Middleware and Architectures for Autonomic and Sustainable Computing	Academic	[29]

Table C.4: Overview of the primary studies.

Title	Authors	Year	Publication Type	Venue	Validation	Reference
A BDI agent-based approach for Cloud Application autonomic management	Xu, L. et al.	2012	Conference	IEEE 4th International Conference on Cloud Computing Technology and Science (CloudCom 2012)		[31]
A review of middleware approaches for energy management in distributed environments	Noureddine, A. et al.	2012	Journal	Software: Practice and Experience		[30]
A Survey on Energy Efficient Server Consolidation Through VM Live Migration	Sekhar, J. et al.	2012	Journal	International Journal of Advances in Engineering & Technology		[43]
A service framework for energy-aware monitoring and VM management in Clouds	Katsaros, G. et al.	2012	Journal	Future Generation Computer Systems	Academic	[22]
An energy aware framework for virtual machine placement in cloud federated data centres	Dupont, C. et al.	2012	Conference	2012 Third International Conference on Future Energy Systems (e-Energy 2012)	Academic	[27]
Using Queuing Theory for Controlling the Number of Computing Servers	Sevalnev, M. et al.	2012	Conference	Proceedings of the 3rd International Conference on Green IT Solutions (ICGREEN 2012)	Academic	[44]
Cloud Engineering is Search Based Optimization too	Harman, M. et al.	2012	Journal	Journal of Systems and Software		[40]
Cloud federation in a layered service model	Villegas, D. et al.	2012	Journal	Journal of Computer and System Sciences		[23]
CompatibleOne: Designing an Energy Efficient Open Source Cloud Broker	Carpentier, J. et al.	2012	Conference	Second International Conference on Cloud and Green Computing (CGC 2012)	Academic	[33]
Green cloud computing schemes based on networks: a survey	Xiong, N. et al.	2012	Journal	IET Communications	Academic	[38]
Towards a Cloud Infrastructure for Energy Informatics	Wu, Z. et al.	2012	Journal	Sprouts: Working Papers on Information Systems	Academic	[37]

Table C.5: Overview of the primary studies (continued).

Strategy	Techniques	Description	Occ.	References
Energy Monitoring	Metering	Power consumption real-time monitoring through external power meters.	6	[33] [19] [20] [22] [29] [24]
	Static Classification	Energy classification of software based upon the power consumption specifications of the hardware components.	3	[25] [19] [35]
	Modeling	Power consumption on-line estimation using predictive models.	4	[26] [18] [27] [29]
Self-Adaptation	Scaling	Software is able to scale down in case of low requests or usage, to save energy.	6	[39] [18] [34] [40] [22] [24]
	Consolidation	In virtualization scenarios, the possibility to regroup VMs sparse among many servers, to reduce the number of active machines.	11	[33] [34] [27] [40] [22] [36] [30] [43] [38] [24] [31]
	Workload Scheduling	Some components of the software architecture are devoted to manage and schedule the workload of the computational units.	18	[25] [39] [26] [18] [20] [35] [40] [41] [28] [22] [36] [42] [29] [43] [44] [38] [24] [31]
Cloud Federation	Energy Brokering	The software architecture exposes their services together with their energy consumption information.	3	[20] [21] [23]
	Service-Adaptation	Switching functional services depending on their energy consumption.	4	[20] [23] [37] [38]

Table D.6: Overview of the identified architectural techniques for energy efficiency.

Strategy	Components	Role	Occ.	References
Energy Monitoring	Energy Dashboard	Provides users or managers with software energy consumption information.	4	[33] [19] [20] [22]
	Energy Database	Stores energy consumption information.	5	[25] [33] [19] [22] [29]
	Energy Indicators	"Rate" or classify software behaviour, or provide real-time metrics upon energy consumption.	4	[26] [27] [20] [22]
	Energy Collectors	Retrieve and collect energy information from hardware or software sensors.	7	[33] [19] [26] [20] [22] [29] [24]
	Energy Communication Bus	Provide a common interface for collectors to the energy database.	3	[33] [19] [22]
	Energy Model	Estimate or predict the power consumption of a software application in real-time.	6	[26] [18] [27] [35] [29] [24]
	Energy Monitor	Monitor the energy consumption of (a part of) the software system.	4	[26] [27] [29] [24]
Self-Adaptation	Adaptation Engine/Optimizer	Find an optimal solution to an objective function modeling the energy efficiency of the system.	12	[25] [39] [26] [35] [40] [41] [28] [22] [36] [42] [24] [31]
	Workload Scheduler	Define, schedule and assign workloads to computational units.	14	[25] [39] [20] [35] [40] [41] [28] [22] [36] [42] [43] [38] [24] [31]
	Scale Unit	A defined set of IT resources that represents a certain scaling level.	1	[34]
	Queue	Organize items (services, VMs, jobs) in different orders of priority according to energy consumption.	3	[18] [42] [44]
	VM Allocator	In virtualized environments, migrate and displace VMs on servers.	13	[25] [33] [34] [18] [27] [40] [22] [36] [30] [43] [38] [24] [31]
	Internal SLA violation checker	Check and ensure the fulfillment of SLAs (NOTE: in this case the checker evaluates the violation of internal services towards external clients).	15	[39] [34] [26] [18] [27] [35] [40] [41] [28] [22] [36] [30] [42] [24] [31]
Cloud Federation	Energy broker	Provides access to energy efficient services.	3	[20] [21] [23]
	Energy Orchestrator	In SOA contexts, switch services in case of relevant differences in their energy efficiency.	3	[23] [37] [38]
	Green Service Directory	Provides a listing of all available services with energy consumption information.	1	[21]
	External SLA violation checker	Check and ensure the fulfillment of SLAs (NOTE: in this case the checker evaluates the violation of external services towards internal clients).	3	[27] [20] [23]

Table E.7: Overview of software components for energy efficiency.