Green Architectural Tactics for the Cloud

Giuseppe Procaccianti*†
*VU University Amsterdam, the Netherlands
†Politecnico di Torino, Italy
g.procaccianti@vu.nl

Patricia Lago
VU University Amsterdam, the Netherlands
p.lago@vu.nl

Grace A. Lewis*‡
*VU University Amsterdam, the Netherlands
‡CMU Software Engineering Institute, USA
glewis@sei.cmu.edu

Abstract—**Energy efficiency is a primary concern for the ICT sector. In particular, the widespread adoption of cloud computing technologies has drawn attention to the massive energy consumption of data centers. Although hardware constantly improves with respect to energy efficiency, this should also be a main concern for software. In previous work we analyzed the literature and elicited a set of techniques for addressing energy efficiency in cloud-based software architectures. In this work we codified these techniques in the form of Green Architectural Tactics. These tactics will help architects extend their design reasoning towards energy efficiency and to apply reusable solutions for greener software.**

I. Introduction

The global energy consumption of data centers and Internet infrastructure is predicted to consume approximately 18% of expected world power capacity by 2030 [1]. As the adoption of cloud computing technologies continues to grow, the need for energy-efficient solutions becomes evident. Nonetheless, cloud-based software holds great potential for energy efficiency (EE). A recent study [2] showed that migrating all business applications in the U.S. to the cloud could reduce their energy footprint by 87%. A previous work [3] started to analyze the cost and energy benefits of data migration to the cloud. However, this transition to the cloud is not an easy task. Cloud-based software must be appropriately designed to address EE, which is typically not the case for traditional business applications. If these applications are abruptly migrated, it is highly likely that the resulting energy waste would significantly outweigh the expected benefits. For this reason, it is time for software engineers to take into account the energy implications of their design decisions. Recent efforts of the software engineering community [4] have stressed the responsibility of software in the search for sustainability. One of the challenges that emerged for Green Software Engineering is finding generally reusable solutions for energy-efficient software design. In previous work we performed a Systematic Literature Review (SLR) [5] to discover software architectural solutions for cloud-based software that addressed EE-related issues. The SLR identified a number of recurring techniques that were potentially reusable in other solutions. In this work we codified these techniques in the from of Green Architectural Tactics. These tactics can be adopted by software architects and developers during the design and development of cloud-based software systems or the refactoring of existing business applications for cloud migration. This contribution will support decision-making when dealing with EE aspects of cloud-based software architectures.

The remainder of the paper is structured as follows. Section II introduces EE as a quality attribute. In Section III the Green Architectural Tactics are presented and described with application examples extracted from the literature. Finally, Section IV discusses the architectural implications of EE and Section V presents some conclusions and anticipates our future research.

II. Energy Efficiency as a Quality Attribute

According to Bass et al. [6], EE is to be regarded as a "system" quality attribute because it is the result of an indirect action of software. However, Bass et al. also argue that the line between "software" and "system" quality attributes is very thin. In the end, even if energy is ultimately consumed by hardware, it is software that determines hardware behavior. In order to provide a clear representation of EE as a quality attribute, we follow the approach adopted by Bass et al. [6] and characterize EE through *quality attribute scenarios*. Each scenario is described in terms of six characteristics:

- *Stimulus.* An event that motivates an EE action.
- *Source of Stimulus.* The entity that triggered the event.
- *Environment.* The set of circumstances under which the scenario takes place.
- *Artifact.* The element of the system that is stimulated by the event.
- *Response.* The action that the system should perform in response to the event.
- *Response Measure.* The metric that determines if the response is satisfactory.

We grouped our Green Architectural Tactics in three categories and formulated a scenario for each category (see Table I). In all the scenarios, the *response measure* is energy consumption values.

III. Green Architectural Tactics

In previous work [5] we identified a set of recurring design solutions, described in the literature, to achieve EE in cloud-based software architectures. In this work, we codified these solutions as *tactics* – that is, "design decisions that influence

TABLE I. QUALITY ATTRIBUTE SCENARIOS FOR ENERGY EFFICIENCY

	Energy Efficiency Scenarios		
Category	*Energy Monitoring*	*Self-Adaptation*	*Cloud Federation*
Stimulus	Request for energy consumption information	Energy consumption alert	Energy consumption alert
Source of Stimulus	Administrator	Energy Monitor	Energy Monitor
Environment	Normal operation	Runtime	Multi-cloud
Artifact	Energy Monitor	Hypervisor	Orchestrator
Response	The Energy Monitor presents the detailed energy consumption information for the data center.	The Hypervisor consolidates the VMs on the less-active servers and then shuts down the idle servers.	The Orchestrator swaps the most energy-consuming services for less energy-consuming services.
Response Measure	Energy consumption values	Energy consumption values	Energy consumption values

the achievement of a quality attribute response" [6]. Each tactic is described in terms of:

- *Motivation:* rationale behind the tactic.
- *Description:* components introduced by the tactic and explanation of their roles.
- *Constraints:* necessary conditions for applying the tactic in an existing software architecture.
- *Example:* previous application of the tactic.
- *Dependencies:* whether the tactic requires other tactics to be applied.

In the following, we describe the identified scenarios for each category, as well as an example of a Green Architectural Tactic.

A. Energy Monitoring

A typical scenario for EE that involves Energy Monitoring is the following: the system administrator of a cloud-based system wants to know the energy consumption of its infrastructure during operations. The Energy Monitor gathers the energy consumption information and presents it to the administrator. The tactics in this category are targeted at monitoring the energy consumption of the cloud infrastructure. These tactics are often combined with tactics from other categories; Self-Adaptation in particular because information from monitoring components is typically used to trigger adaptive mechanisms.

Example Tactic – Energy Modeling:
Motivation. In order to implement self-adaptive mechanisms it is necessary to have near-real-time energy consumption information. This enables the modification of software behavior according to how much energy the system is actually consuming. When metering systems are unavailable, the Energy Modeling tactic is a viable option.
Description. The Energy Modeling tactic enables a dynamic estimation of power consumption values through predictive Energy Models. These Models are embedded in Energy Indicators, components that provide the power consumption estimation to the rest of the software architecture, e.g. to enable energy–aware behavior. Typically, Energy Models are built through regression analysis based on software runtime metrics, i.e. resource usage (CPU, disk, memory) [7].
Constraints. The limitation of this tactic lies in the accuracy of the software Energy Models. To date, many models and tools are available to estimate software energy consumption but their accuracy varies greatly based on the selected hardware platform. In addition, not all hardware resources are good predictors of energy consumption; identifying the best predictors

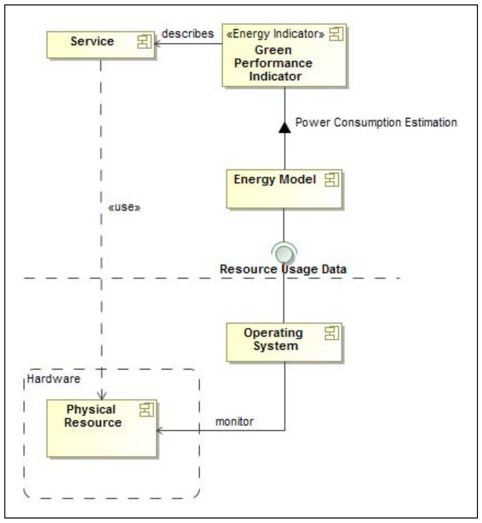

Fig. 1. Example of the Energy Modeling tactic.

is still an issue for the researchers in the field [8].
Example. A prototype showing the application of this tactic is provided by de Oliveira et al. [9]. The context is a Service-Oriented Architecture (SOA) applied to a cloud infrastructure. As shown in Figure 1, for each service of the SOA, the Operating System of each physical node provides service-related Resource Usage Data (in the case of the example, CPU, memory and disk [9]). A linear Energy Model retrieves this data and estimates the power consumption impact of each service. The estimation is modeled into a Green Performance Indicator (GPI), of type Energy Indicator. Each GPI describes a service in terms of EE.

B. Self-Adaptation

The Self-Adaptation scenario for EE starts from the Energy Monitor that reports an alert of excessive energy consumption while the system is not fully loaded. In response, the Cloud Hypervisor (i.e. the Virtual Machine Monitor [10]) migrates some of the VMs to less-loaded servers so that it can shut down the resulting idle servers.
Tactics in this category implement mechanisms that modify runtime software configurations for the specific purpose of lowering energy consumption. In cloud-based environments Self-Adaptation mostly concerns the configuration, deployment, and workload of Virtual Machines (VMs).

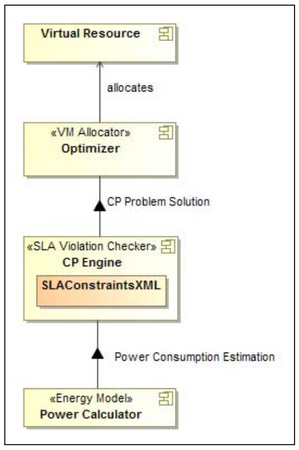

Fig. 2. Example of the Consolidation tactic.

Example Tactic – Consolidation:
Motivation. On-demand resource provisioning is an important feature of cloud-based environments. For example, in cloud application server provisioning[1] creating new VM instances may provide additional flexibility and help to perform load balancing among servers. This is called *horizontal* scaling (or scaling *out*). This operation, however, may easily lead to inefficient usage of physical resources if the density of VMs across the physical servers is not accurately managed in low-request phases. Indeed, the Consolidation tactic concentrates the VM instances on the minimum number of servers needed. Powering down the unused servers will evidently increase the EE of the cloud-based software.
Description. The main component of the Consolidation tactic is the VM Allocator, the software component responsible for live VM migration. This component can be (a part of) the Hypervisor, as in the Adaptation Engine in the Scaling tactic. The SLA Violation Checker is needed as well to check the fulfillment of service-level objectives after VM migrations.
Constraints. Consolidation must take place at runtime. This means that VMs must be represented in a format that allows them to be seamlessly migrated from one location to another, along with their context, workload, and metadata. This may introduce high network traffic and security risks.
Example. Dupont et al. [11] provide a sample implementation of the Consolidation tactic, depicted in Figure 2. The Power Calculator, of type Energy Model, provides a power consumption estimation to the CP Engine, of type SLA Violation Checker. The CP Engine formulates a constraint programming problem using the constraints extracted from the SLAs in XML format (SLAConstraintsXML). The CP engine solves the problem and the Optimizer (of type VM Allocator) produces a VM allocation scheme applying the solution to the Virtual Resources.
Dependencies. The presence of the Power Calculator indicates a dependency from the Energy Modeling tactic: as shown in Figure 2, the Power Calculator is an instance of an Energy Model.

[1]http://www.enterprisenetworkingplanet.com/netos/article.php/3753836/Practical-VM-Architecture-How-Do-You-Scale.htm, last visited on October 4th, 2013.

C. Cloud Federation

The Cloud Federation scenario for EE is the following: the Energy Monitor notifies about an excessive energy consumption arising from a service, which is a composition of multiple cloud services. The Service Orchestrator then tries to swap some services in the service composition by searching in a Green Service Directory for services that consume less energy than those currently being used.
A cloud federation is a multi-cloud environment that can be defined as "[*a platform that*] comprises services from different providers aggregated in a single pool" [12]. Cloud Federation tactics allow cloud-based software systems to "lease" or "negotiate" cloud services from multiple providers based on energy consumption information.

Example Tactic – Service-Adaptation:
Motivation. The main benefit of the Cloud Federation paradigm is the possibility to select services among different providers. Through Energy Monitoring tactics, we are able to know the energy consumption of services. This enables cloud-based software systems to discover services that are more energy-efficient than those currently in use. The Service-Adaptation tactic describes how Cloud platforms should switch to these more energy-efficient services.
Description. Two components realize the Service-Adaptation tactic. The first component is the Energy Orchestrator that discovers energy-efficient services that fulfill a certain task and eventually performs the registration of those services with the system. This operation has to be authorized by the second component, the SLA Violation Checker, which ensures that the new services meet the service-level objectives required by the system. This component is similar to its analog in the Self-Adaptation tactics, but instead of checking the SLOs that *internal* services have to fulfill, it checks if *external* services meet the SLOs required by the system.
Constraints. The Service-Adaptation tactic assumes centralized cloud service orchestration. This creates some disadvantages in terms of flexibility because it concentrates all service orchestration logic in a single point.
Example. Villegas et al. [13] illustrate an example of the Service-Adaptation tactic in a federated cloud architecture. In their view, the Service-Adaptation is performed at the Software-as-a-Service (SaaS) layer: whenever a service request to the federated cloud cannot be fulfilled with the required service level or is too costly in terms of energy, it is forwarded to another federated cloud provider. As shown in Figure 3, the SaaS Broker, of type Energy Orchestrator, negotiates the usage of a Green Service with other cloud providers. The Reputation of the cloud provider (of type SLA Violation Checker) determines if the provider meets the required service-level objectives. The Reputation is based on the SLA violation rate of the provider.
Dependencies. As implied by the tactic description, Service-Adaptation depends on Energy Monitoring tactics in order to retrieve the energy information of services.

IV. Discussion

The Green Architectural Tactics presented in this work were explicitly formulated with reusability in mind. For this reason, we kept to a minimum the constraints that a tactic may impose on the general software architecture. When necessary,

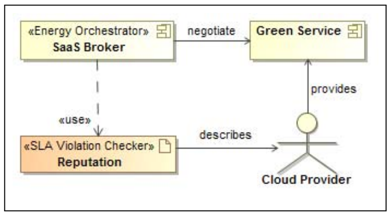

Fig. 3. Example of the Service-Adaptation tactic.

TABLE II. ENERGY EFFICIENCY TRADEOFFS INTRODUCED BY GREEN ARCHITECTURAL TACTICS

Tactic	*Quality Attribute*	*Rationale*
Consolidation	Availability	During VM migration some services may not be available.
	Security	Live VM migration over the network requires to transfer application code, metadata and workloads, making them vulnerable to attacks.
Energy Modeling	Modifiability	Energy Connectors are component-specific and therefore must be reimplemented if the architecture changes.
Service-Adaptation	Flexibility	The orchestrator concentrates all service composition logic in a single node.

we made them explicit. For example, the Service-Adaptation tactic assumes the presence of a service orchestration mechanism; most of the Energy Monitoring tactics introduce a centralized Energy Database; Self-Adaptation tactics assume a high degree of decoupling between the virtual and the physical infrastructure. If these tactics are meant to be applied to an existing cloud-based system, software architects should consider whether these assumptions are compatible with the current architecture. Sometimes, Green Architectural Tactics cannot be adopted in isolation: when introducing them in a software architecture, they might require other tactics to be adopted as well. In the previous section, we made such dependencies explicit. It is relevant to point out that the occurrence of a combination of tactics does not always imply a dependency. This suggests a deeper relationship between the tactics that we will further explore in our future research. Furthermore, our tactics introduce tradeoffs between EE and other quality attributes, summarized in Table II. Along with the scenarios provided in Section III, this initial trade-off analysis contributes to the identification of EE as a quality attribute. It is still under discussion to what extent EE and other sub-characteristics of environmental sustainability might influence traditional quality requirements. In our ongoing and future work, we are investigating this topic.

V. CONCLUSIONS

In cloud computing, energy efficiency aspects have to be addressed from a software architecture perspective. In this work, we provide a set of reusable design solutions, codified as *tactics*, to support the design and development of cloud-based energy efficient software. In order to help their understanding and adoption, each of our Green Architectural Tactics is presented with an example of its application extracted from literature. In addition, in this contribution we describe energy efficiency as a software quality attribute and analyze its architectural impact in terms of assumptions and tradeoffs. In the future, we foresee the inclusion of energy efficiency in a comprehensive software quality model. This will be the focus of our future research: we are exploring the impact of energy efficiency on other software quality attributes and we will investigate guidelines and methods for assessing the energy efficiency of existing software systems.

ACKNOWLEDGMENT

This work has been partially sponsored by the European Fund for Regional Development under project MRA Cluster Green Software and the Department of Defense under Contract No. FA8721-05-C-0003 with Carnegie Mellon University for the operation of the Software Engineering Institute, a federally funded research and development center (DM-0000704).

REFERENCES

[1] C. Preist and P. Shabajee, "Energy Use in the Media Cloud: Behaviour Change, or Technofix?" in *IEEE Second International Conference on Cloud Computing Technology and Science (CloudCom)*. IEEE, 2010, pp. 581–586.

[2] E. Masanet, A. Shehabi, L. Ramakrishnan, J. Liang, X. Ma, B. Walker, V. Hendrix, and P. Maantha, "The energy efficiency potential of cloud-based software: A u.s. case study," Laurence Berkeley National Lab, Berkeley, California, Tech. Rep., June 2013.

[3] Q. Gu, P. Lago, and S. Potenza, "Delegating data management to the cloud: a case study in a telecommunication company," in *International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems (MESOCA)*, vol. 7. IEEE Computer Society, sep 2013, p. 8.

[4] P. Lago, R. Kazman, N. Meyer, M. Morisio, H. A. Müller, F. Paulisch, G. Scanniello, B. Penzenstadler, and O. Zimmermann, "Exploring initial challenges for green software engineering: summary of the first GREENS workshop, at ICSE 2012," *SIGSOFT Softw. Eng. Notes*, vol. 38, no. 1, pp. 31–33, Jan. 2013.

[5] G. Procaccianti, S. Bevini, and 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*, vol. 1. Shaker Verlag GmbH, 2013, pp. 291–299.

[6] L. Bass, P. Clements, and R. Kazman, *Software architecture in practice*, 3rd ed. Addison-Wesley, 2012.

[7] S. Reda and A. N. Nowroz, "Power modeling and characterization of computing devices," *Foundations and Trends in Electronic Design Automation*, vol. 6, no. 2, pp. 96–216, 2012.

[8] G. Procaccianti, L. Ardito, A. Vetrò, and M. Morisio, "Energy Efficiency in the ICT - Profiling Power Consumption in Desktop Computer Systems," in *Energy Efficiency - The Innovative Ways for Smart Energy, the Future Towards Modern Utilities / InTech*. Helwan: Prof. Moustafa Eissa, 2012, pp. 353–372.

[9] 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.

[10] G. J. Popek and R. P. Goldberg, "Formal requirements for virtualizable third generation architectures," *Communications of the ACM*, vol. 17, no. 7, pp. 412–421, 1974.

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

[12] T. Kurze, M. Klems, D. Bermbach, A. Lenk, S. Tai, and M. Kunze, "Cloud federation," in *The Second International Conference on Cloud Computing, GRIDs, and Virtualization (CLOUD COMPUTING)*, 2011, pp. 32–38.

[13] D. Villegas, N. Bobroff, I. Rodero, J. Delgado, Y. Liu, A. Devarakonda, L. Fong, S. Masoud Sadjadi, and M. Parashar, "Cloud federation in a layered service model," *Journal of Computer and System Sciences*, vol. 78, no. 5, pp. 1330–1344, 2012.