Green Information Systems for Sustainable IT

C. Cappiello, M.G. Fugini, B. Pernici, P. Plebani[1]

Abstract In this paper, we present the approach to green information systems adopted in the Green Active Management of Energy in IT Service centres (GAMES) Project. The goal of GAMES is to develop methodologies, models, and tools to reduce the environmental impact of information systems at all levels, from application and services to physical machines and IT plants. This paper proposes models and methods for the analysis and reduction of the energy consumption associated with information systems.

Introduction

Information management has become pervasive in modern society and the impact of IT on energy budget is becoming more and more significant. While the focus of research in IT has been in getting more and more performing and reliable systems, the analysis of the impact of information systems from the point of view of energy consumption has been lacking. Research activity is mainly focusing on power management in large data centres or on technical characteristics of devices from the point of view of power consumption [13].

In this paper, we overview the approach to green information systems studied in the Green Active Management of Energy in IT Service centres (GAMES) EU Project. GAMES considers the environmental impact of resources involved in the whole life cycle of an information system, from design to run time and maintenance, in organizations. The goal of GAMES is to develop methodologies, models, and tools to reduce the environmental impact of such systems, reducing energy consumption and energy losses at all levels of the information system structure, from application and services, to physical machines and IT plants. The focus of the paper is on methods to analyze energy losses in and in actions that can be undertaken to save energy, such as redundancy elimination at different levels, from applications to services, middleware and data and processing infrastructures. To this aim, a set of *Green Performance Indicators* (GPI) is presented, together with a set of Green Actions which can be undertaken on various system components (CPU, memory, applications, and so on) once an energy leakage has been discovered through monitoring. Such Green Actions allow one to (partially) remove that

[1] Dip. Elettronica e Informazione, Politecnico di Milano, piazza da Vinci, 32, I-20133 Milano, Italy e-mail:{cappiell,fugini,pernici,plebani}@elet.polimi.it

energy loss by, for example, reducing redundancies of data and processes, by using the memory in slow mode, or by improving the rate of Application Server usage.

Actually, while redundancy is needed to ensure Quality of Service (QoS) [11, 12] of a system in operation, such redundancy can be potentially eliminated or reduced in the system after its use. General examples are data replicated in many archives, devices which are dimensioned for peek time elaboration of information, useless data such as spam mails. GAMES focuses on green IT in information systems, with the aim of developing an "IS purifier" approach, in analogy to what is being done for cleaning water for a sustainable environment.

GAMES Approach to Green Information Systems

The GAMES approach proposes guidelines for designing and managing service-based information systems along the perspectives of energy awareness. The approach focuses on the following two aspects, with the ultimate goal of developing a new systematic scientific discipline in the area of Green Computing:

a) the *co-design of energy-aware information systems* and their underlying services and IT service centre architectures in order to satisfy users, context, and QoS requirements, addressing energy efficiency and controlling emissions. This will be carried out through the definition of suitable Green Performance Indicators (GPI) able to evaluate if and to a what extent a given service and workload configuration will affect the carbon footprint emissions' levels;

b) the *run-time management of IT Service Centre energy efficiency*, which will exploit the adaptive behavior of the system at run time, both at the service/application and IT architecture levels, considering the interactions of these two aspects in a overall unifying vision.

The integrated approach to green IT undertaken in GAMES focuses on IT use and management, and on an energy-saving design and management of application and data resources of the information system. The primary goal of the project is to develop a systematic scientific discipline in the area of green information systems, in particular relying on Web services technology, which is suitable to support *adaptivity* to different system states and necessities in front of energy saving policies.

Services become a major asset since they allow developers and managers to set the basis for building flexible adaptable systems able to react to variable operation conditions. Specifically, in GAMES we define a green life cycle (from system design to maintenance) which allow, in an integrated view, to develop adaptive, self-healing, and self-managing systems able to reduce energy consumption, also considering impact factors on the environment.

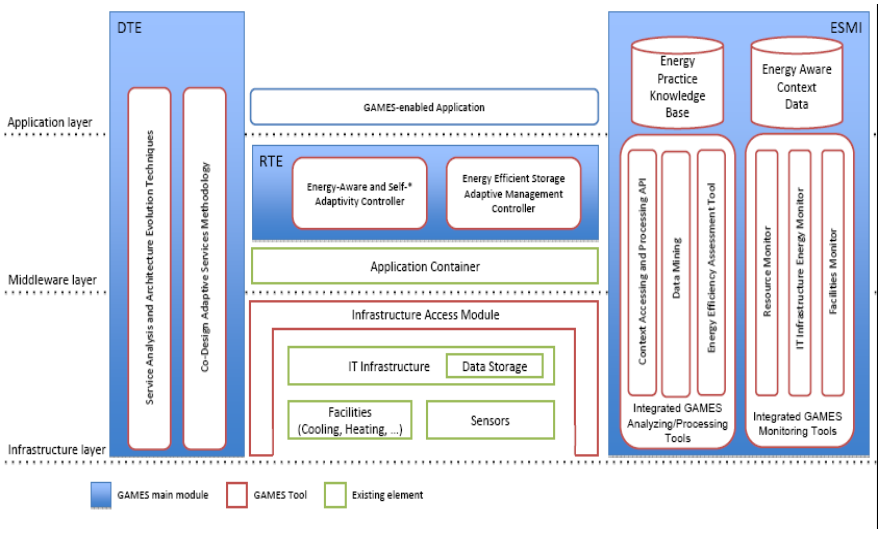

Figure 1 - GAMES general architecture

Figure 1 shows the GAMES architecture. The modules are organized in three areas: the DTE (Design-Time Environment), the RTE (Run-Time Environment) and the ESMI (Energy Sensing and Monitoring Infrastructure) and on three layers, namely: Infrastructure, Middleware, and Application.

At the Infrastructure layer, all the tools directly dealing with the physical machines and devices are included. All the physical devices used for monitoring the IT infrastructure and the environment (such as sensors, power meters etc.) are assumed to exist in advance and they are not implemented in the project. An Infrastructure Access Module provides, through an API, the operations needed to query or configure the service centre IT infrastructure, environmental sensors and facilities.

At the Middleware layer, assuming the existence of a standard application container, the Run-Time Environment module (RTE) enhances it in order to provide the adaptation services required by GAMES applications. Standard application containers refer to middleware that offer a set of shared services (e.g., transactional, security) that the applications, which have been deployed on it, can use. In order to be able to adapt at run-time, GAMES applications must be designed according to the GAMES energy-aware application co-design methodology provided by the DTE module. This methodology consists of a set of models describing how the system should react in case some GPI are not satisfied or are not going to be satisfied at run-time. As a consequence, besides the usual functional and non-functional description, the GAMES application descriptor also includes a set of rules defining adaptation strategies and the relationships among the layers composing the GAMES architecture. Application container takes care of function-

al and non-functional descriptors, while the adaptation strategies are exploited by the RTE module to select the proper adaptation actions to enforce the predefined GPI. The RTE can be seen as an extension of current usual application containers by providing the services that enact the adaptation.
The ESMI module contains a set of tools for: (i) collecting the run-time energy/performance data (such as Nagios or Hyperic) and (ii) analyzing/processing the run-time energy/performance collected data (by using Weka/SpagoBI data mining tools and Pellet reasoner). The run-time energy/performance data (referred as context data) collected and processed by the ESMI tools refer to: IT infrastructure energy/performance data, environmental data and the state of the GAMES enabled application running on the service centre.

At run-time, the RTE uses the context data to take adaptation decisions using a context model. The context model elements are instantiated at run-time with the context data captured by the ESMI tools. These instances are stored in the Energy Aware Context Data repository. The context model instances are processed to determine the service centre energy/performance state and to infer new context information that is relevant for the decision process. Context data are made available to all interested GAMES architecture modules through the Context Accessing and Processing API.

The Energy Practice Knowledge Base stores the energy/performance knowledge obtained by executing mining algorithms on the historical data collected by the ESMI monitoring tools.

The GAMES architecture run-time adaptation modules are organized according to the four MAPE stages of a self adaptive system[2].

[2] M. Salehie, L. Tahvildari, "Self-adaptive software: Landscape and research challenges", ACM Tran. on Aut. and Adaptive Systems,
ISSN:1556-4665, 2009

At the Application layer, the *GAMES-enabled applications* are located, designed according to the DTE methodology. A GAMES-enabled application is a process composed by activities usually defined in terms of their functional and non-functional requirements. These requirements, for a given activity, can be fulfilled by a set of services that run on virtual machines. At run-time, instances of processes, activities, and services are defined in terms of their execution states. Along with the usual definition of a process in terms of activities, data flow/control flow and KPIs, a GAMES-enabled application will be also defined in terms of information about GPIs and adaptation strategies to be enacted in case the objectives of one or more GPIs are not fulfilled.

Methodology for Green Applications

Starting from the GAMES architecture illustrated above, the methodology we are studying in the project considers that an application, or a service, performs in a "green way" if it delivers its expected results:

- saving energy consuming, for instance, less processor and/or less memory and storage and/or less I/O
- according to the user QoS requirements.

Moreover, we consider an application as a flow of activities executed by means of services. A methodology towards green services has to evaluate factors such as the intensity of use of Processor, Memory, Storage, and I/O peripherals, as well as the application flow given by the application structure and its activities/services and data.

For example, given the sample application described by the flow in Figure 2, the factors related to energy consumption are the following.

- Activities/Services need IT resources and consume power
- Data are read/written on storage and transferred to I/O
- Data objects (volatile) are read/written in Memory
- The application has a structure with a design-time defined workflow, and additional information (e.g., branching and failure probabilities)
- QoS parameters are annotations provided aside (e.g. response time, performance, duration, costs, data accuracy, security)

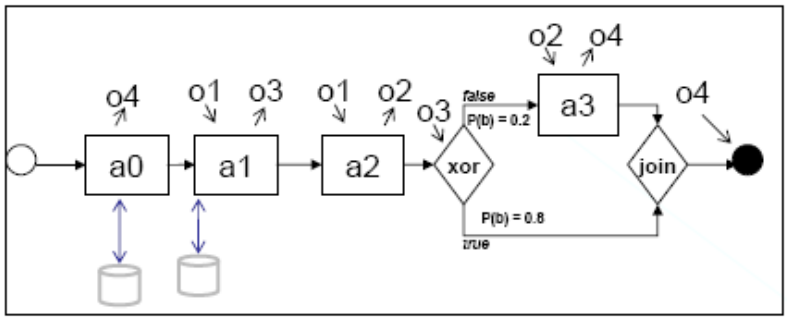

Fig. 1 Sample application flow with activities, read/written objects, databases, and annotations about branching probabilities on the xor split

An annotation at design-time which can help to reveal energy usage is the branching probability P(b) associated with every outgoing edge of a control activity, representing the probability of executing the corresponding branch. In Figure 2, branching probabilities associated with the xor control node are P(b) = 0.8 for the true branch and P(b) = 0.2 for the false branch. Information about branching probabilities can be provided by the designer at design time (or added to the application definition by the process management system). Determining these probabilities is a hard and time-consuming task (see [14] for an approach to compute branching probabilities of instance subgraphs). Nevertheless, branching probabilities are useful for energy consumption computation in that, if it is known that, after the xor, the probability of executing a_3 is low, the machine where such service is made available can stay turned off or run in idle mode most of the time. Moreover, branching probabilities are saved on the system log and monitored, to update the application definition accordingly. Using this approach, the designer does not need to determine probabilities at design time but can rather add them after various runs.

The aim of the methodology is to design energy-aware adaptive applications through energy *annotations*. Annotations characterize applications from their energy consumption view point, so that designers, by observing several runs of a same application in different cases through different instantiations, can annotate the application with branching probabilities and other structure-dependent information, such as used data and activities, but also with data regarding how much of machine resources the process needs.

Annotations can be used to describe the typology of the application (e.g., whether it is a data intensive or a CPU intensive application, for instance but also in which application domain it operates, e.g., finance or materials production). The typology brings about different resource needs. This information can be enriched with additional annotations that can be inferred from *log data* collected along runs of various instances of the same application. In fact, logs are able to

provide data about the usage of processor, memory, storage, and I/O peripherals and thus a first characterization of energy needs [15]. We keep annotation as independent as possible of a given physical configuration of machines and storage devices, in order to be able to estimate the way an application uses the processor, the memory, the storage, and the I/O peripherals, and, through many runs, derive information about the possibility of that application to run also in a less resource consuming way.

In particular, through annotations, an application can be made *adaptive with respect to energy consumption* if the amount of needed resources can be adapted on the basis of energy and QoS requirements. Adaptivity is defined as a kind of recovery/compensation process that is required in case the application does not achieve, or is not going to achieve, the required GPIs. Strategies could have different degrees of complexity going from the substitution of a single activity, to the re-design of the whole application. Strategies can also affect the infrastructure elements (such as consolidation or dynamic power management).

We consider that two levels are available to analyze the energy consumed (and possibly wasted) by an application:

- The *infrastructure level* - we observe the IT resource (processor, disks, I/O devices, etc.) usage, as well as other parameters regarding energy consumed by cooling systems, buildings and so on. In this paper we concentrate on IT resources only.
- The *application level* - we observe the application activities and their energy consumption. The user poses some QoS constraints on the application (e.g. total cost of execution) and these should be respected.

Adaptivity regards how an application can be adapted to run in different modes (e.g., low processor usage, slow disk, etc.). If the infrastructure is wasting say processor utilization (the CPU results under-occupied for a process due to over-dimensioned allocation), then we say we have an *energy leakage*. In general, we have to determine which parameter(s) of the 4 is actually wasting energy and, then, adapt the process running mode to avoid energy leakage.

Adaptivity regards also the application level: an energy leakage can be observed when services result as unused, or data are redundant, or when a service is located on a flow branch with low branching probability. Then, the application can be adapted for example by removing the unused service, or by removing redundant data. The issue here is to keep the required values for QoS met for the user (the application level is hence user-requirements centered).

We can also combine the two levels to observe *global application energy consumption* (and possible waste). For example, if an application can run also in “less data storage” mode, respecting the data QoS requirements in terms of response time, we can adapt it to use less data storage and to clean redundant data.

Annotating the application *at design time* means to enrich the application flow with energy-aware information regarding the fact that this application can run also with less storage consumption. However, if the application in low mode does not

meet the QoS requirements (*QoS violation*), the same application is annotated as to be run in high data mode only.

The annotation at design time is based on process mining techniques foreseen in the project. The idea is that through several runs, several instances of the application can be observed giving an application profile in terms of energy usage.

The methodology has to foresee also a set of strategies for detecting, correcting, and adapting an application in the elements which are *wasting energy at run-time*. Let us give an example, where adaptation is performed at run-time.

Example: Strategy of *Adaptation through Substitution of Activities*

Substitution is a strategy to be used when running activities are considered as definitively unavailable or temporarily not appropriate since they violate some energy or QoS constraints. To perform or complete the application execution, a substitution strategy allows changing the activity by finding a service that provides the same operations. Suppose that several equivalent candidate activities are available. In such case, energy and QoS constraints can be the driver for the service selection. In case of a substitution due to service failure, we aim at *spending the same or lower energy of the correct execution*.

$$E(a_i) \geq E(a_i)_{tf} + E(a_{sub}) + E(a\text{-}new_{ij})$$

where $E(a_i)_{tf}$ is the energy consumed by a_i until the failure time *tf* , $E(a_{sub})$ is the energy spent in the operations performed to implement the substitution (e.g., compensation of the failed activity) and $E(a\text{-}new_{ij})$ is the energy associated with the execution of the *j*-th substitute activity.

In general, a strategy in the methodology has to consider energy leakage in detail. We give a discussion in what follows, considering that applications are annotated about their energy needs.

Reasoning about Energy Leakage

Leakage of energy signals unused, or badly used resources at application middelware, and infrastructure level. Here we consider the application level only. Energy leakage can be evaluated by comparing different configurations for activities, using their annotations. For instance, if in one instance the process execution consumes more energy than in another process instance, we can define the *leakage as the amount of wasted resources derived from the difference between the two states*.

One way of dealing with leakage is to define an approach to energy awareness embedded in information systems based on GPI which are the drivers of adaptation to energy requirements. GPI allow energy consumption to be monitored during the information system life cycle leading for instance to removing unwanted services and data, tuning the IT resource use, or improving the quality of the development phases.

GPI are layered at strategic, tactical and operational levels [5]. At the strategic level [3], GPI drive high-level decisions about a system organization in terms of used human resources, impact on the environment (e.g., in terms of logistics, IT procurement, acquisition of new energy saving software and hardware), outsourcing of non-core services, guidelines for system development according to eco-related laws and regulations such as EU Code of Conduct for Data Centres 2010[3], Energystar[4], United Nations Global Compact[5], Scottish Environmental Protection Agency[6], and so on. GPI at the tactical level denote how the service system will consume less energy if its development phases are enhanced through the use of mature platforms (which are likely to produce more energy-aware systems) [10], through improvement of the system quality in terms of service delivery versus customers' expectations and in terms of less complexity of the overall service interfaces. Other GPI at this level regard decommissioning of unused services and data, so recovering system space and resources by cleaning areas and resources.

At the operational level, we define GPI for monitoring the usage of IT resources (processor, memory, I/O and storage [2]). We consider these four factors as a characterization of energy needs [9], and keep GPI independent of physical configurations of machines and storage devices, in order to be able to estimate the way a service system uses the resources, and, through several runs, derive information about the possibility for that system to run in a lower resource-consuming way.

Conclusions and Future Work

In this paper, we have presented the basics of the GAMES approach and methodology that enables evaluating energy consumption and efficiency in applications starting from the analysis of the characteristics of the activities composing the application.

The concept of annotation of an application toward energy-awareness can be pursued further in several directions. First, the application schema can be related to the infrastructure for its execution. Assuming that in an energy-aware context the infrastructure itself is developed with Green IT criteria [16], application activities are executed on Virtual Machines. The mapping of activity characteristics and virtual machines configurations has to be further investigated, as well as its relation to energy consumption in different configurations. The annotation presented in this paper should be extended to consider not only unitary elements, but

[3]EU Code of Conduct for Data Centres 2010.
http://re.jrc.ec.europa.eu/energyefficiency/html/standby_initiative_data_centers.htm

[4] Energystar http://www.energystar.gov/

[5] United Nations Global Compact http://www.unglobalcompact.org/

[6] Scottish Environmental Protection Agency http://www.sepa.org.uk/

also a richer activity profile, for instance data volumes, disk access characteristics, and exploiting the data dependencies defined in the process to improve data management toward energy efficiency. In the EU Games project, further work will also consider modelling the data centres with all its business processes, IT infrastructure and facilities, and to mine from execution data significant correlations that can be analyzed to identify energy leakages.

References

1. A. Arsanjani, S. Ghosh, A. Allam, T. Abdollah, S. Gariapathy, and K. Holley. SOMA: A Method for Developing Service-oriented Solutions. IBM Systems Journal, 47(3):377-396, 2008.
2. David Brown and Charles Reams. Toward energy-efficient computing. Communications of the ACM, 53(3), March 2010.
3. F. David. Strategic Management. Merrill Publishing Company, 1989.
4. Brian Doherty. Green IT Report 2008. Technical report, Cap Gemini, 2008.
5. Jan van Bon, Arjen de Jong, Axel Kolthof, Mike Pieper, Eric Rozemeijer, Ruby Tjassing, Annelies van der Veen, and Tienke Verheijen. IT Service Management - An Introduction. Van Heren Publishing, Netherlands, 2007.
6. Toby Velte, Anthony Velte, and Robert Elsenpeter. Green IT: Reduce Your Information System's Environmental Impact While Adding to the Bottom Line. McGraw-Hill Companies, Sept. 2008.
7. John Viega. Cloud computing and the common man. IEEE Computer, 42:106-108, 2009.
8. Mathias Weske. Business Process Management: Concepts, Languages, Architectures. Springer Berlin 2007.
9. Satoshi Sekiguchi, Satoshi Itoh, Mitsuhisa Sato, and Hiroshi Nakamura. Service aware metric for energy efficiency in Green data centers. 2009. http://www.iea.org/work/2009/standards/Sekiguchi.pdf
10. Felipe Alberato, Sustainable Software Engineering, Carnegie Mellon Silicon Valley Tech Report, 2004.
11. Liangzhao Zeng, Boualem Benatallah, Marlon Dumas, Jayant Kalagnanam, and Quan Sheng. Quality Driven Web Services Composition. In Proceedings of the 12th World wide Web Conference (WWW), 2003.
12. Anbazhagan Mani and Arun Nagarajan. Understanding Quality of Service for Web Services. http://www-128.ibm.com/developerworks/library/ws-quality.html, 2002.
13. Chase, J. S., Anderson, D. C., Thakar, P. N., Vahdat, A. M., and Doyle, R. P. (2001). Managing energy and server resources in hosting centers. SIGOPS Oper. Syst. Rev., 35(5):103–116.
14. Zuoxian Nie, Xin-hua Jiang, Jian-cheng Liu, Hai-yan Yang, "Performance Analysis of Generalized Well-formed Workflow," Eight IEEE/ACIS International Conference on Computer and Information Science (ICIS 2009), Shanghai June 2009, pp.666-671
15. Brown, D.J., Reams, C.: Toward energy-efficient computing. Communications of the ACM 53(3) (March 2010), pp. 50-58
16. Velte, T., Velte, A., Elsenpeter, R.: Green IT: Reduce Your Information System's Environmental Impact While Adding to the Bottom Line. McGraw-Hill Companies (Sept. 2008)