The Green Grid'5000: Instrumenting and Using a Grid with Energy Sensors

Marcos Dias de Assunção, Jean-Patrick Gelas, Laurent Lefèvre, and Anne-Cécile Orgerie

Abstract Targeting mostly application performance, distributed systems have constantly increased in size and computing power of their resources. The power supply requirements of these systems have increased in a similar fashion, which has raised concerns about the energy they consume. This paper presents the Green Grid'5000 approach[1], a large-scale energy-sensing infrastructure with software components that allow users to precisely measure and understand the energy usage of their system. It also discusses a set of use-cases describing how an energy instrumented platform can be utilised by various categories of users, including administrators, Grid component designers, and distributed application end-users.

1 Introduction

Large-scale distributed systems have become essential to assist scientists and practitioners in addressing a wide range of challenges, from DNA sequence analysis to economic forecasting. Driven mostly by application performance, these systems have constantly increased in size and computing power of their resources. The power supply requirements of these systems have increased in a similar fashion, which has raised concerns about the electrical power they consume and fostered research on improving their efficiency.

Marcos Dias de Assunção · Jean-Patrick Gelas · Laurent Lefèvre · Anne-Cécile Orgerie
INRIA RESO (UMR CNRS, INRIA, ENS, UCB), 46 allée d'Italie 69364 Lyon Cedex 07 - France, e-mail: marcos.dias.de.assuncao,jean-patrick.gelas,laurent.lefevre,annececile.orgerie@ens-lyon.fr

[1] Some experiments of this paper were performed on the Grid'5000 platform, an initiative from the French Ministry of Research through the ACI GRID incentive action, INRIA, CNRS and RENATER and other contributing partners (http://www.grid5000.fr)

A range of techniques can be utilised for making computing infrastructures more energy efficient, including better cooling technologies, temperature-aware scheduling [8], Dynamic Voltage and Frequency Scaling (DVFS) [11], and resource virtualisation [3]. To benefit from these techniques and devise mechanisms for curbing the energy consumed by large-scale distributed systems [5], it is important to instrument the hardware infrastructure. The instrumentation can provide application and system designers with the information they need to understand the energy consumed by their applications and design more energy efficient mechanisms and policies for resource management. However, monitoring large-scale distributed systems such as Grids remains a challenge. Although various solutions have been proposed [13, 2, 12, 6] for monitoring the usage of resources (*e.g.* CPU, storage and network) to improve the performance of applications, only a few systems monitor the energy usage of infrastructures [9, 10].

This paper presents the Green Grid'5000 approach, a large-scale energy-sensing infrastructure with software components that allow users to precisely measure and understand the energy usage of their system. When architecting the energy-sensing infrastructure and deciding on the required data management techniques, the following goals were considered:

- Increase users' awareness on the energy consumed by their applications;
- Improve the design of user applications;
- Obtain data to advance Grid middleware;
- Help managers in implementing power management policies; and
- Create an energy-data repository for further studies.

The rest of this paper is organised as follows. Section 2 presents the deployed energy sensing hardware infrastructure, whereas Section 3 describes the design of software components. Section 4 presents some concrete use-cases benefiting from the Green Grid'5000 solutions. Conclusion and future work are presented in Section 5.

2 Instrumenting a Distributed Infrastructure with Energy Sensors

The Grid'5000 platform [1], depicted in Figure 1, is an experimental testbed for research in distributed computing which offers 5000 processors geographically distributed across 9 sites in France (*i.e.*Bordeaux, Grenoble, Lille, Lyon, Nancy, Nice, Orsay, Rennes, and Toulouse), all linked by dedicated high-speed networks.

This platform can be defined as a highly-reconfigurable, controllable and monitorable experimental Grid. Its utilisation is specific: each user can reserve a number of nodes on which she deploys a system image; hence, the node is entirely dedicated to the user during her reservation. The Resource

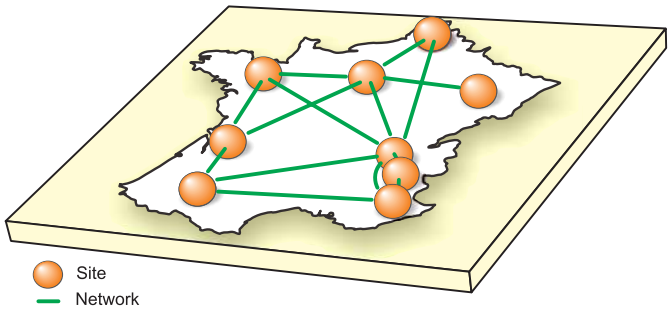

Fig. 1 The Grid'5000 infrastructure in France.

Management System (RMS) used by Grid'5000 is called OAR [4]. OAR is an open-source batch scheduler that provides a simple and flexible interface for reserving and using cluster resources.[2] Users can submit best-effort jobs, which are started by OAR when resources are available, and advance reservations where users specify the desired start time of their requests. Figure 2 illustrates one of the interfaces provided by the Grid'5000 API, showing the status of the platform. The API also gives users access to information about each node, such as CPU, memory, network, disk and swap usages. However, to obtain this information a daemon must run on each reserved node. As running this daemon is optional because it can interfere with user experiments, the information may not be available for all reservation requests.

Although Grid'5000 is different from a production infrastructure, the concerns surrounding its energy consumption are similar to those of production environments. Therefore, solutions proposed to tackle challenges in Grid'5000 can fit both experimental and production infrastructures.

The energy-sensing infrastructure[3] has been deployed on three Grid'5000 sites (*i.e.* Lyon, Grenoble, and Toulouse) monitoring in total 160 computing and networking resources. The infrastructure comprises a large set of wattmeters manufactured by OMEGAWATT.[4] Each wattmeter is connected to a data collector via a serial link. A dedicated server stores the gathered data and generates live graphs of energy usage.

In the Lyon site, where the whole infrastructure is monitored, the wattmeters periodically measure the energy consumed by 135 nodes and a router. Four Omegawatt boxes where required; one 48-port box, two 42-port boxes and one 8-port box. Each box works as a multi-plug measurement device. For example, in the case of the 8-port box, 8 physical nodes can be plugged into it and the energy data is transmitted to a data collector (*i.e.* a remote server) via a serial link (*i.e.* RS232). Hence, in this site the energy data-collector

[2] `http://oar.imag.fr/index.html`

[3] The infrastructure is financially supported by the INRIA ARC GREEN-NET initiative and the ALADDIN/Grid'5000 consortium.

[4] http://www.omegawatt.fr/gb/index.php

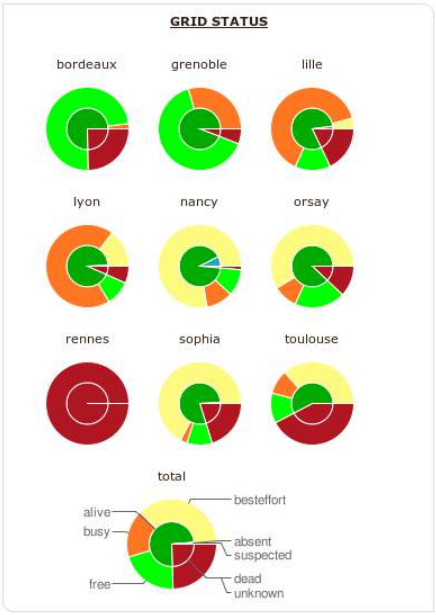

Fig. 2 Example of information provided by Grid'5000 API.

server is connected to four boxes through four different serial links. As the energy data collector does not have four serial ports, it uses USB-RS232 converters. Another problem faced with serial links, was that the node racks are not next to each other, but in front of each other. Thus, two boxes are connected to the energy data collector via 6-meter long USB cables; cables must go under the floor. Although USB cables have a theoretical limit of 5 meters, above which distance they do not operate well, we have not observed failures during the ten months that this infrastructure is in operation.

On the data collector, a daemon for each box is responsible for collecting the energy data. At each second, this daemon sends a data request in an hexadecimal code with a CRC to the box to ask for the energy consumption value of all the resources monitored by that box. The response is an hexadecimal code with a CRC in the form of "packets", each packet containing the instantaneous power consumption of six nodes. The software responsible for communicating with the wattmeters for obtaining the data was not provided with the boxes. We had to design it by ourselves for collecting at every second the measurements of 135 nodes including CRC checking, hexadecimal decoding and storage of these values. The data gathering process should not exceed one second.

As the goal of instrumenting the Grid infrastructure was to provide energy consumption information to different groups of users and system administra-

tors, several data management techniques had to be applied. For example, the energy data is stored in three different ways:

- Raw data: that includes a timestamp indicating when a measurement was performed (there is one log file per resource).
- Last values: one file (in memory file systems) per resource and the value is over-written at each second.
- Round-Robin Databases[5]: one file per monitored node.

The data collection and storage process for the Lyon site is illustrated in Figure 3. The usage of each type of storage for the different categories of users is detailed in the following sections.

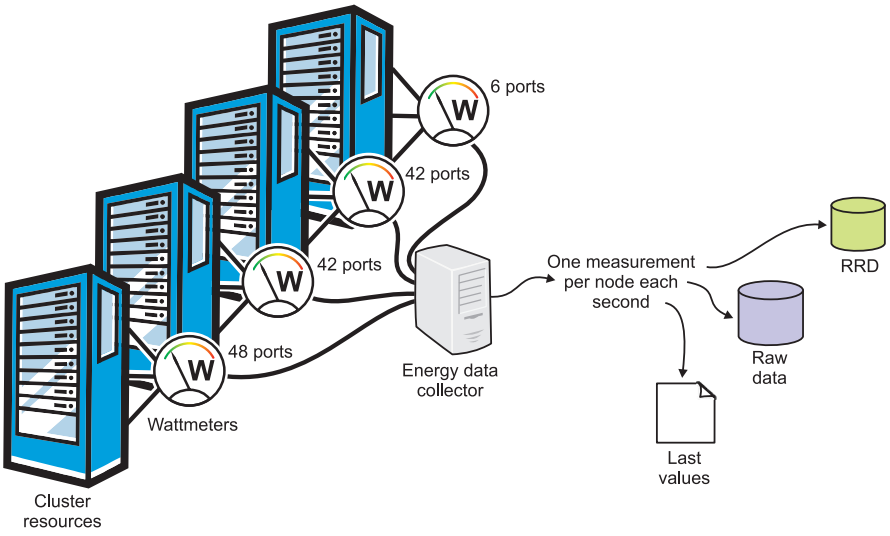

Fig. 3 Energy data collector deployed on the Lyon site.

As each category of user has different requirements, the frequency of measurement is important and must be calibrated depending on what one wants to observe. For administrators that want to observe global energy trends, a few measurements per hour or per day can suffice. Users who want to understand usage and applications impact on energy consumption, demand several measurements per minute. For middleware designers who want to understand peak-usage phenomena (during a system deployment phase for example), multiple measurements per minute are mandatory. Moreover, in this case the data must be stored to allow one to evaluate trends and patterns that can aid the improvement of middleware tools. In the Green Grid'5000

[5] RRDtool is the OpenSource industry standard, high performance data logging and graphing system for time series data (`http://oss.oetiker.ch/rrdtool/`).

context, one live measurement (in Watts) is performed each second for each monitored resource with a precision of 0.125 Watts. These measurements are displayed lively and stored in specific repositories. One year of energy logs for a 150-node platform corresponds to approximately 70 GB of data.

These energy logs can be correlated with the reservation logs (provided by OAR) and the resource usage logs provided by the Grid'5000 API in order to have a more complete view on the energy usage.

3 Displaying Information on Energy Consumption

Measuring energy consumption and storing the data are only part of the process when the goal is to reduce the amount of electricity utilised by an infrastructure. Ideally, the obtained information on energy consumption should drive optimisations in the resource management systems; inform users of the cost of their applications in terms of electricity spent; and possibly offer means to raise users' awareness and adapt their behaviour so they can make more energy-aware choices when executing their applications. In addition, the information should be presented to users in a way they can measure the impact of their choices in terms of energy consumption when they run their applications differently from the manner they are used.

Towards achieving these goals, we have developed a set of tools for visualising the data collected from instrumenting the Grid platform. This section describes a graphical interface, termed as *ShowWatts*, for displaying the measurements in near real-time, and a list of Web pages and graphs for viewing historical data on energy consumption. The first interface allows users to view the instant energy consumption of the whole platform and check the immediate impact of executing an application on a set of resources; whereas the Web pages and graphs enable the analysis of energy consumption over larger periods and the identification of patterns and application behaviours.

3.1 ShowWatts: Live Display

ShowWatts comprises a Java Swing application for displaying the energy consumption of the instrumented platform in nearly real-time (see Figure 4), and a module responsible for aggregating the energy consumption information and transferring it to the node where it is displayed. It works on the client-server model: a daemon is launched on the server side and another is launched by the client; the client daemon uses the "last values" described in Section 2. These two processes communicate through an SSH tunnel. The server side part is optimised to minimise the data volume sent: only the values that differ from the previous transfer are sent.

From a system administrator's point of view, this interface can be used to monitor the whole platform or the consumption of specific resources. The administrator can select the resources in which she is interested and view their share of the overall energy consumption. In addition, it is possible to view other statistics such as a rough amount of money spent with electricity.

Additionally, the application is customisable and can be changed to display information on other platforms. This is achieved by implementing the module that aggregates the consumption information and changing two XML files; one that describes general configuration and display preferences and another that specifies the monitored resources. For demonstrative purposes, we have implemented a reservation scheduler in python, which shows the impact in energy consumption when switching machines off and on, and aggregating reservation requests.

3.2 Portable Web-based Visualisation

As mentioner earlier, in addition to displaying the instant energy consumption, visualisation tools must provide means whereby users and system administrators can analyse past consumption. By visualising the energy consumed by individual resources or groups of resources over larger periods, administrators and users can better know the behaviour of their platform, design more energy-efficient allocation policies and investigate possible system optimisations.

We have developed a set of scripts that automatically generate Web pages and graphs for displaying the energy consumption of individual resources and groups of resources. Figure 5 shows an example of Web page containing a list of graphs with measurements taken from a Grid'5000 site. These graphs are updated every minute so that the user has an up-to-date view of the platform in terms of energy consumption.

3.3 Collecting and Providing Energy Logs

Measuring the power consumption of a Grid is the first step to understand the energy usage of the platform, while the following step is to collect and store this data and give users access to it. As discussed in Section 2, to collect the energy measurements, we have developed a software to interact with the wattmeters and to store the received information in different formats: RRD databases and text files.

RRD databases offer the advantage of having a fixed size. They use average values for generating graphs of long periods. They are used to have an overview of the energy platform. It provides different views of the node's

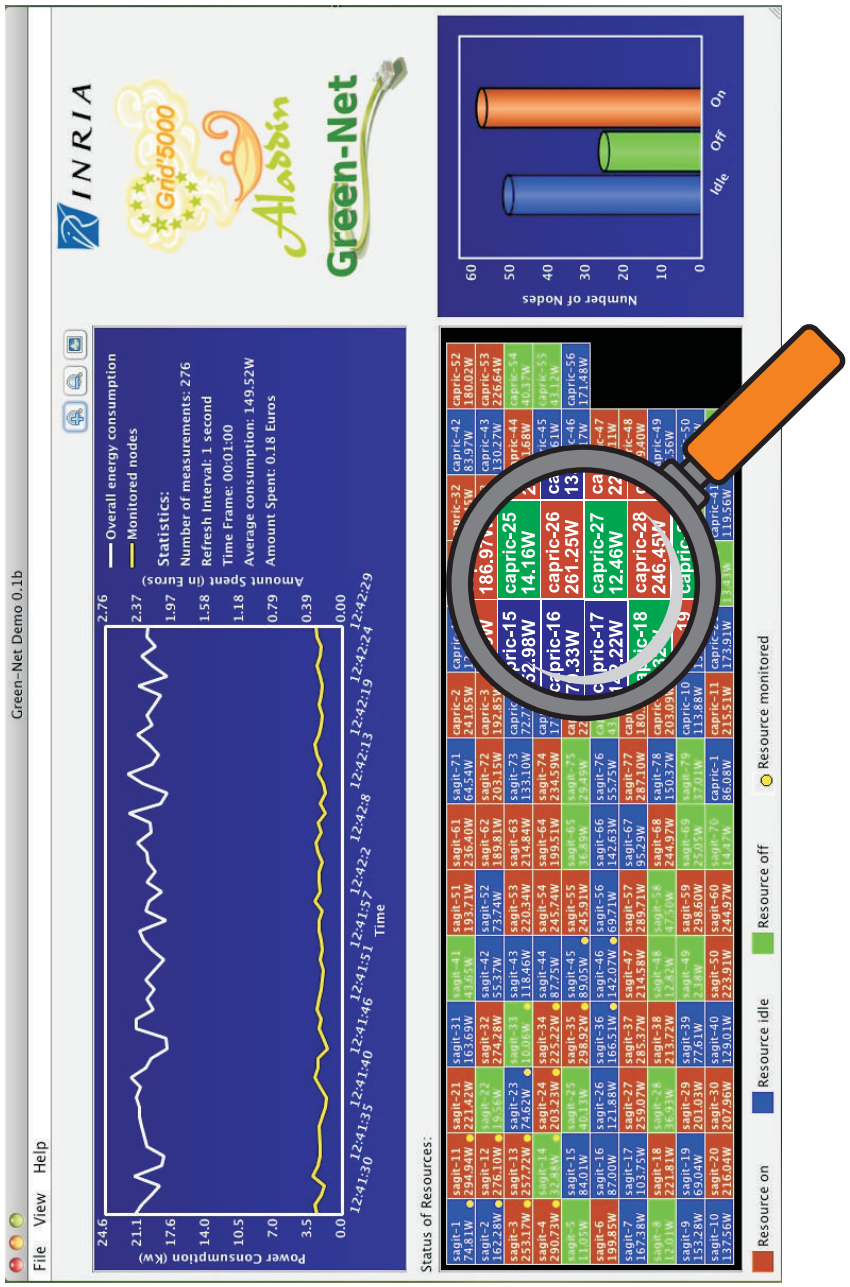

Fig. 4 Screenshot of ShowWatts graphical interface.

Fig. 5 Web interface for energy usage visualisation.

energy consumption (*i.e.* per hour, per day, per week, per month and per year). Each view presents maximum, minimum and average values. The current value per node (in Watts) is also displayed and updated every second (by an ajax script) as well as the global value for the whole site. These tools are particularly suitable for administrators. They provide a quick overview of the whole platform with an in-depth access to the most recent values for each node. An example of usage for this kind of energy log is given in Section 4.4.

These aggregated views, however, are not precise enough to understand the link between running an application on a node and the energy it consumes. Hence, text files are used to store each measurement (*i.e.* for each node at each second with a timestamp). These text files are archived and compressed each week in order to reduce the disk space required; this allows us to keep all the logs.

While collecting and storing energy logs is important, the main goal is to provide these logs to users, so that they can see the impact of their applications on energy consumption of the platform. This may increase users' awareness about the energy consumed by large-scale infrastructures. Therefore, the monitoring Website includes *log-on-demand* features. As shown in Figure 7, the Web portal entails a Web page where a user queries the energy consumption of servers by specifying the period, the time increment (*i.e.* value in seconds between two measurements) and the nodes. These tools are specially designed for the users and the middleware developers who want to precisely profile their applications in terms of energy usage and to correlate the energy data with resource (*e.g.* CPU, memory, disk, network, etc.) utilisation to view the impact of their programming decisions on energy con-

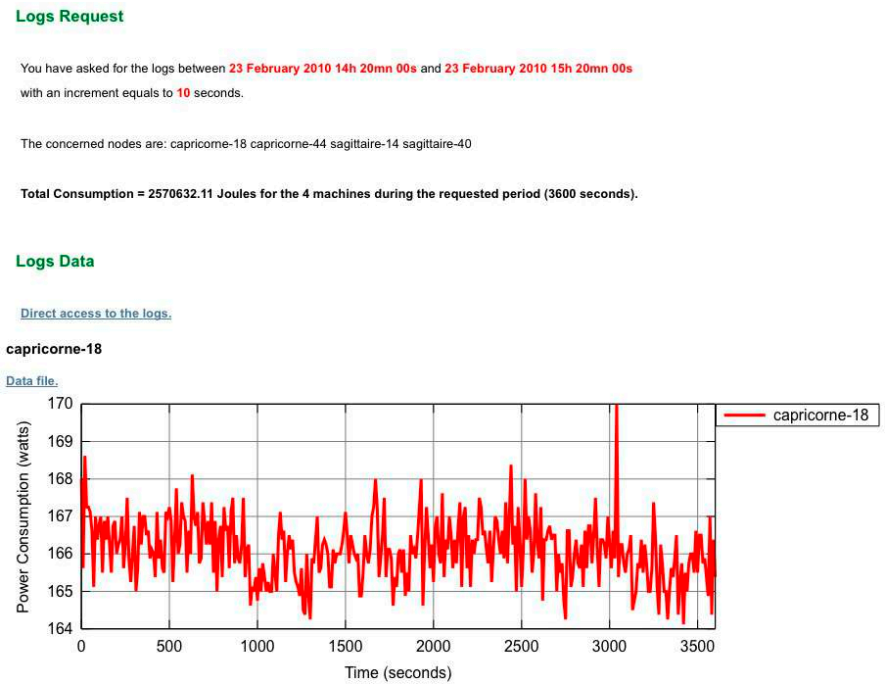

Fig. 6 Web page displaying the result of an energy-log request.

sumption. Examples of such applications and utilisations of these tools are given in Sections 4.1, 4.2 and 4.3.

The architecture responsible for collecting, storing and exposing the logs is summarised in Figure 8. The logs and graphs are available for all Grid'5000 users and administrators. Figure 6 presents the typical result of a log request. First, the global information of the request is given, such as the global consumption of the requested nodes during the given time frame. Moreover, for each node, the energy graph is provided to show the energy profile of the user's request.

For each request, we provide a directory (with a permanent address) that contains:

- a log file for each requested node with all the collected measurements during the time period requested (with the requested increment between two measurements);
- an SVG graph for each requested node plotting its energy profile;
- a log file with the global consumption of each requested node during the requested time period and the total consumption of the requested nodes; and
- a .tar.gz file containing all the files above.

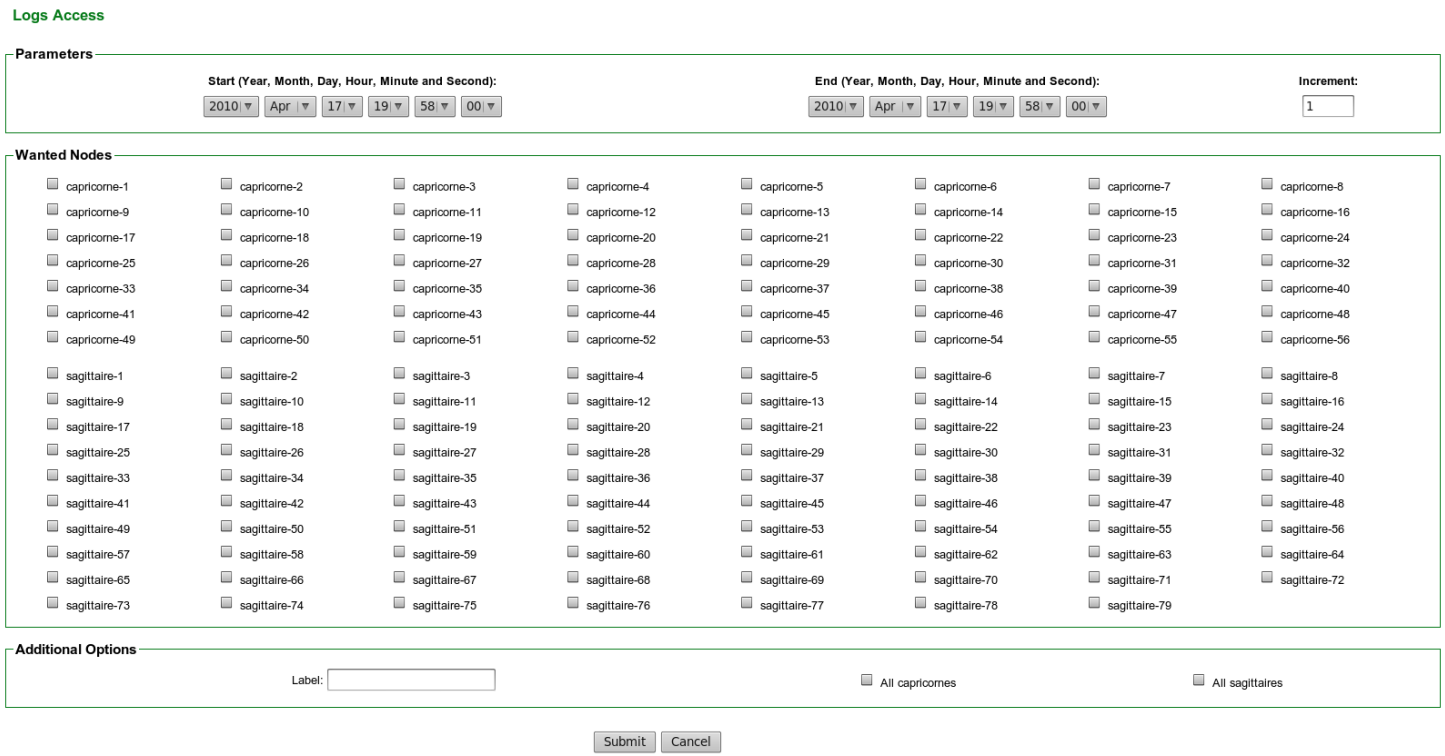

Fig. 7 Web Interface for energy-log requests.

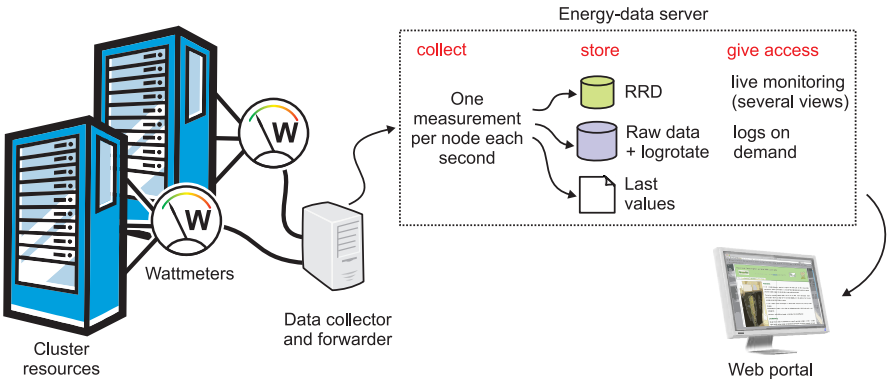

Fig. 8 Overall architecture to collect, store and expose the energy data.

4 Green Grid'5000 Usage Examples

This section presents some examples on how the energy consumption data can be utilised by different types of users.

4.1 Energy Profiling of Applications

The energy monitoring architecture is often utilised by Grid'5000 users to study the energy profile of their applications. Figure 9 illustrates the behaviour of the typical life-cycle of a user application on a Grid'5000 node (Sun Fire V20z: 2 CPUs of 2.4GHz). The different phases of the application correspond to:

- the boot of the node that lasts for approximately 120 seconds and is characterised by power consumption peaks;
- then, an idle period during which the node still has a high energy consumption even if it does nothing (around 240 Watts);
- an *hdparm*[6] experiment which represents intensive disk-accesses (243 Watts on average);
- an *iperf*[7] experiment which makes intensive use of the network (using UDP, it consumes 263 Watts on average but also uses CPU);
- a *cpuburn*[8] which fully loads a CPU (267 Watts on average);

[6] *hdparm* is a command line utility for Linux to set and view the IDE hard disks hardware parameters.

[7] *Iperf* is a commonly-used network tool that can create TCP and UDP data streams to measure TCP and UDP bandwidth performance.

[8] *cpuburn* is a tool designed to heavily load CPU chips in order to test their reliability.

- a *stress*[9] which fully uses all the CPUs and also performs I/O operations (277 Watts on average);
- a halt during which a small power consumption peak can be observed; and finally
- the node is off and still consumes around 10 Watts.

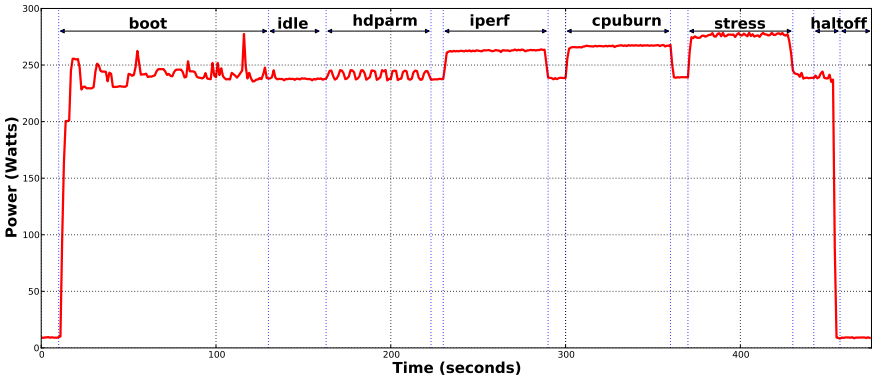

Fig. 9 Energy consumption during typical node applications.

The *logs on demand* tool has been used to create Figure 9. As one can observe, the tool provides detailed information and a summary; the experiment has lasted 474 seconds, and during that period the node consumed 111,296.34 Joules. Energy profiling of user applications can greatly increase their awareness as well as help programmers to improve the energy efficiency of their applications.

4.2 Improve Distributed Applications

As end-users are able to profile the energy usage of their applications, it is possible to design more efficient distributed applications by applying some "green" programming concepts. As an illustrative scenario, a programmer designs an MPI client-server application as depicted in Figure 10, where 3 processes run concurrently:

- one server is assigning tasks to two clients,
- client 1 is computing small tasks (S Tasks), and
- client 2 is computing XL Tasks (1 XL task = 2 S tasks).

At the first attempt, the programmer designs a basic application where the server assigns tasks to clients in a synchronous way. This application is

[9] *stress* is a tool to impose load on and stress test systems.

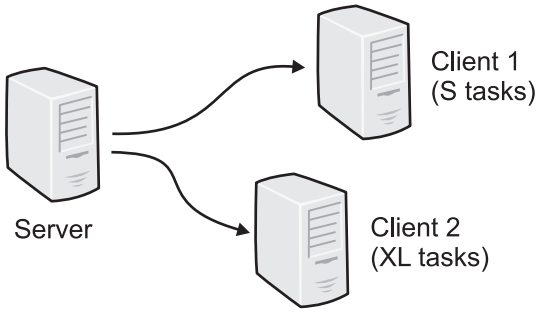

Fig. 10 Basic MPI application scenario.

neither optimal in terms of performance nor energy efficient. As tasks are heterogeneous, some periods of inactivity are observed on client 1 and energy is wasted (Figure 11). The client spends long periods of inactivity with an idle consumption of 179 Watts. By using the Green Grid'5000 infrastructure, the end-user is able to detect these idle periods and re-design her application.

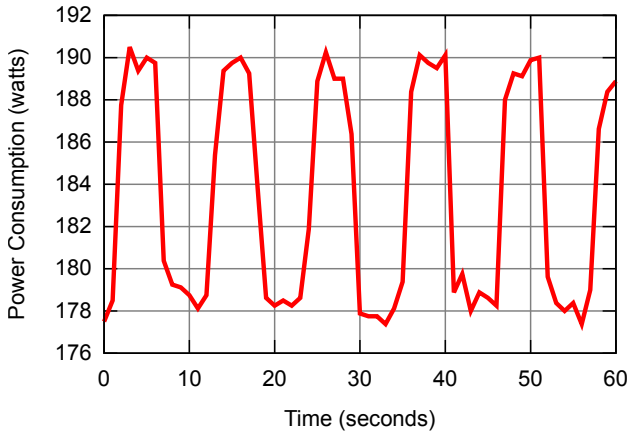

Fig. 11 Power consumption of client 1 during the non-energy efficient MPI application.

The programmer then decides to use asynchronous MPI communication. As soon as the client has finished to compute its tasks, it sends the results back to the server, which is then able to send new tasks to the client. We can observe that this more aggressive approach is better in terms of energy and performance as it reduces inactivity periods (Figure 12). In one minute, the client can compute more tasks than in the previous example. By profiling applications, designers can explore ways to improve their energy efficiency.

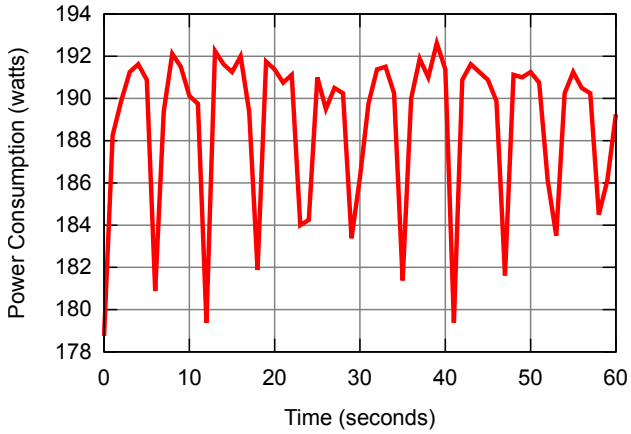

Fig. 12 Power consumption of client 1 during the energy-efficient MPI application.

4.3 Improving the Energy Efficiency of Grid Middleware

As discussed beforehand, recent computer hardware and operating systems have offered a range of techniques to manage the power drawn by servers. The challenge posed to middleware designers is how to benefit from these techniques for making distributed systems more energy efficient, hence reducing both their energy footprint and the costs associated with managing a Grid infrastructure. Tackling this challenge requires an understanding of how users utilise the infrastructure and how their applications behave in terms of resource usage and energy consumption. Previous study on the energy consumption of Grid'5000 has shown that substantial energy savings can be achieved if users accept to change the start or finish times of their reservations, so that they can be aggregated, creating free windows during which unused servers can be switched off [9]. Additional savings can be made when analysing the energy consumption data and crossing it with information from other system components such as the RMS or scheduler.

We utilise the "interactive" advance reservations of Grid'5000 as an example to illustrate how Grid middleware can be improved when information on energy consumption of servers and resource utilisation logs are made available to middleware developers. Under an interactive reservation, a user reserves a group of servers on which she deploys an environment that contains the whole operating system customised to her application. Figure 13 shows the energy consumed by a server of the Lyon site during part of a reservation that starts as 10am on a working day. The noisy measurements obtained after 1900 seconds are typical of an environment-deployment phase during which the disk image is copied to the node that is later rebooted with the new operating system. This figure shows that during the period that pro-

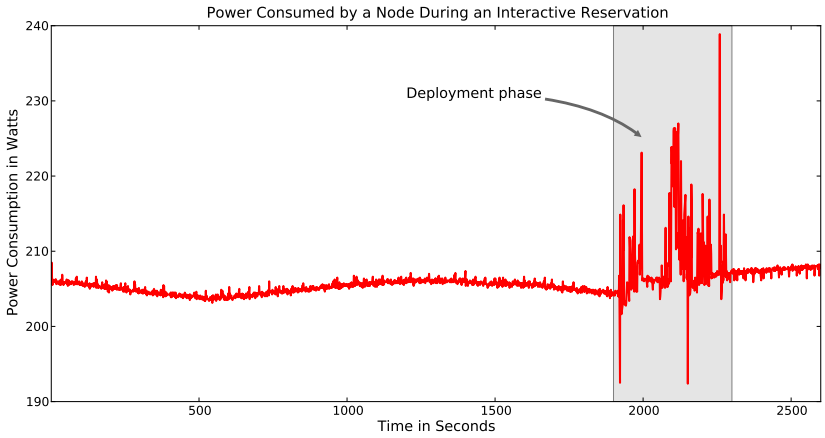

Fig. 13 Energy consumed by a server during part of an interactive reservation.

ceeds the deployment, the server was not utilised, hence wasting resources and energy. If mechanisms for identifying and predicting these behaviours are incorporated into middleware design, the unused servers can be switched off, hence minimising resource wastage and consequently improving the energy efficiency of the infrastructure.

4.4 Administrator's Use Case

Collecting and analysing energy consumption information can aid system administrators: to evaluate the power management techniques offered by the hardware, observe the impact of different policies to curbing the energy consumption of the infrastructure, and guide capacity planing decisions. However, there is only so much that energy consumption data alone can provide. As illustrated in the example above, it is important to cross the data on energy consumption with information from other system components such as cluster schedulers – *e.g.* information on node failures, job arrivals, job scheduling – and information on resource utilisation – *e.g.* information on CPU, memory, storage and network usage.

Figure 14 presents the energy in KWh per day consumed by the Grid'5000 site in Lyon. The blue line shows the resource utilisation according to the site scheduler (*i.e.* OAR) [4]; the utilisation indicates the percentage of reserved nodes, and hence does not imply that CPUs, storage or network resources were used by reservations at the same rate. Although this is a simple example, it illustrates the type of information required by managers to evaluate the efficiency of the infrastructure and identify correlations between resource usage and energy consumption. Managing and analysing this data can provide

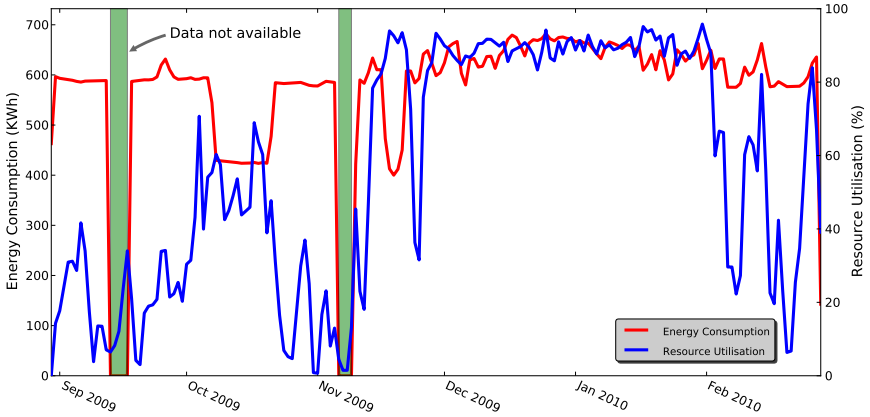

Fig. 14 Energy consumption and utilisation of nodes over six months.

system administrators with hints on how to provision and manage resources more efficiently.

5 Conclusion and Future Work

This paper presents the steps in instrumenting and monitoring the energy consumption of an experimental Grid. As observed, having a configurable sensing infrastructure is one of the basic components mandatory for designing energy efficient software frameworks. The Green Grid'5000 is the first available large-scale platform whose energy is monitored per plug every second. Current research on energy efficiency in large-scale distributed systems is made possible by the instrumented platform [9, 7].

As of writing of this paper, the energy sensing infrastructure has been in operation for over 10 months. Several techniques have been applied to obtain, store and manage the data resulting from monitoring the energy consumed by the infrastructure. The different manners in which the data has been stored and reported aim to serve the needs of several categories of users, including application developers, middleware designers, system administrators and infrastructure managers. In addition to energy consumption data, obtaining and analysing information from multiple sources – *e.g.* schedulers and resource utilisation – is essential to understanding the usage of the infrastructure and applications' behaviours. This analysis can help system designers and managers to identify how to devise and implement techniques for reducing the cost of energy consumption. It also allows them to evaluate the return of investment of deploying the sensing infrastructure itself. At present, instrumenting a platform is expensive and its cost can surpass the

savings achieved by striving to improve the energy efficiency of the overall infrastructure.

As future work, we intend to deploy energy sensors in several Grid'5000 sites and monitor the consumption of both computing and network equipments. In addition, we are in the process of creating a repository where the data on energy consumption will be made publicly available to the research community.

References

1. Cappello et al, F.: Grid'5000: A large scale, reconfigurable, controlable and monitorable grid platform. In: 6th IEEE/ACM International Workshop on Grid Computing, Grid'2005. Seattle, Washington, USA (2005)
2. Andreozzi, S., De Bortoli, N., Fantinel, S., Ghiselli, A., Rubini, G.L., Tortone, G., Vistoli, M.C.: Gridice: a monitoring service for grid systems. Future Generation Computer Systems **21**(4), 559–571 (2005). DOI http://dx.doi.org/10.1016/j.future.2004.10.005
3. Barham, P., Dragovic, B., Fraser, K., Hand, S., Harris, T., Ho, A., Neugebauer, R., Pratt, I., Warfield, A.: Xen and the art of virtualization. In: 19th ACM Symposium on Operating Systems Principles (SOSP '03), pp. 164–177. ACM Press, New York, USA (2003). DOI http://doi.acm.org/10.1145/945445.945462
4. Capit, N., Costa, G.D., Georgiou, Y., Huard, G., n, C.M., Mounié, G., Neyron, P., Richard, O.: A batch scheduler with high level components. In: Cluster computing and Grid 2005 (CCGrid05) (2005). URL `http://oar.imag.fr/papers/oar_ccgrid05.pdf`
5. Da Costa, G., Gelas, J.P., Georgiou, Y., Lefèvre, L., Orgerie, A.C., Pierson, J.M., Richard, O., Sharma, K.: The green-net framework: Energy efficiency in large scale distributed systems. In: HPPAC 2009 : High Performance Power Aware Computing Workshop in conjunction with IPDPS 2009. Roma, Italy (2009)
6. Gu, J., Luo, J.: Reliability analysis approach of grid monitoring architecture. Annual Conference ChinaGrid **0**, 3–9 (2009). DOI http://doi.ieeecomputersociety.org/10.1109/ChinaGrid.2009.28
7. Lefèvre, L., Orgerie, A.C.: Designing and evaluating an energy efficient cloud. The Journal of SuperComputing **51**(3), 352–373 (2010)
8. Moore, J., Chase, J., Ranganathan, P., Sharma, R.: Making scheduling "cool": Temperature-aware workload placement in data centers. In: USENIX Annual Technical Conference (ATEC 2005), pp. 5–5. USENIX Association, Berkeley, CA, USA (2005)
9. Orgerie, A.C., Lefèvre, L., Gelas, J.P.: Save watts in your grid: Green strategies for energy-aware framework in large scale distributed systems. In: 14th IEEE International Conference on Parallel and Distributed Systems (ICPADS). Melbourne, Australia (2008)
10. Singh, T., Vara, P.K.: Smart metering the clouds. IEEE International Workshops on Enabling Technologies **0**, 66–71 (2009). DOI http://doi.ieeecomputersociety.org/10.1109/WETICE.2009.49
11. Snowdon, D.C., Ruocco, S., , Heiser, G.: Power Management and Dynamic Voltage Scaling: Myths and Facts. In: Proceedings of the 2005 Workshop on Power Aware Real-time Computing (2005)
12. Truong, H.L., Fahringer, T.: Scalea-g: A unified monitoring and performance analysis system for the grid. Sci. Program. **12**(4), 225–237 (2004)
13. Zanikolas, S., Sakellariou, R.: A taxonomy of grid monitoring systems. Future Generation Computer Systems **21**(1), 163–188 (2005). DOI http://dx.doi.org/10.1016/j.future.2004.07.002