Towards a novel Smart and Energy-Aware Service-Oriented Manager for Extreme-Scale applications

Mohammed el Mehdi Diouri, Olivier Glück, Laurent Lefèvre

ENS Lyon/LIP Laboratory (ENS, INRIA, University of Lyon, CNRS)

San José, June 5th, 2012

mehdi.diouri@ens-lyon.fr

1st Workshop on Power Grid Friendly Computing, co-located with IGCC'12, San José, 5-8 June 2012

Outline

Context

Supercomputer = machine built from a collection of computers performing tasks in parallel.

An important growth of performance: a factor of 1000/10 years.

The most powerful supercomputer is K-Computer (Top500): more than 700,000 cores and able to perform 10 PFlop/s.

A wide range of scientific applications:

- manufacturing: design of cars and aircraft
- environment: prediction of tsunami and seisms

Context and Motivations

IESP (USA), EESI (Europe): roadmaps to Exascale in 2018.
Exascale supercomputer: more than 10^{18} Flops.

Main challenges at the Exascale:

- How to build these supercomputers ?
- How to provide them energy ?
  - major energy consumers.
  - irregular and dynamic energy consumption.
- How to program applications running on them ?
  - power/energy consumption: several dozens of MW.
  - resilience and fault tolerance: many failures per day.
  - data management: ExaBytes of data transferred.

Motivations

Exascale infrastructures:
Major energy consumers with irregular energy consumption.

Dialog with the energy provider.

Prediction of the energy consumption of the different services.

Service = application portion that can be extracted and that performs a specific function
(scattering algorithm, checkpointing protocol).

Motivations

Interactions between the various actors.

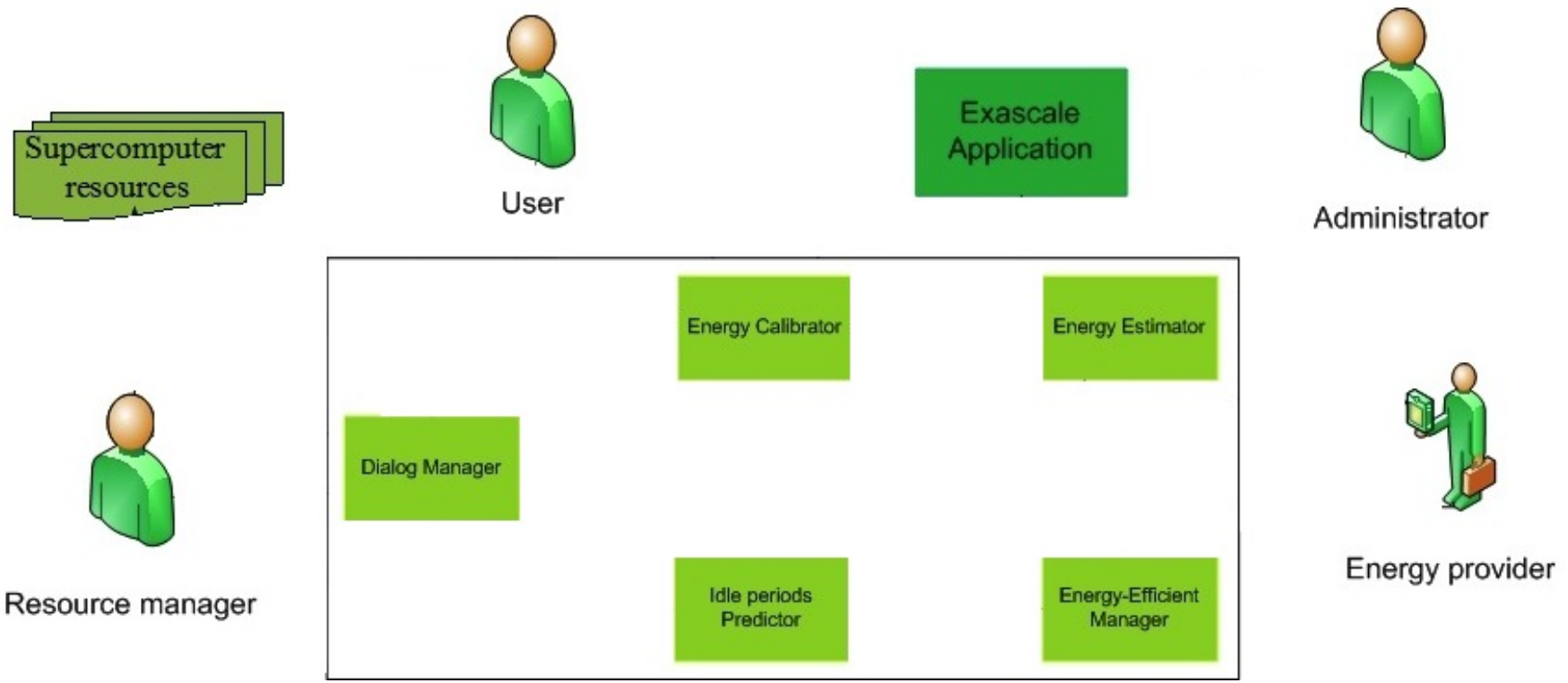

Figure: Towards SEASOMES

Challenges - Fault Tolerance

An exascale supercomputer will typically gather millions of cores.

They will experience various kind of faults many times per day.

To reach exascale application termination, fault tolerance is mandatory.

Two main categories of protocols:
uncoordinated and coordinated protocols: checkpointing/restart:

- with message logging in uncoordinated protocols
- with process synchronization in coordinated protocols.

2 services: Checkpointing and Restart.

Challenges - Data Management

Data management embeds the following services:

- Data scattering;
- Data gathering;
- Data locating;
- Data visualization.

Exascale applications will involve large volumes of data: exabytes of data exchanged.

Significant improvements in data management are necessary.

Challenges - Power efficiency

Another limiting factor to the future growth of HPC.

Including infrastructure installation and maintenance costs.

Most energy efficient: IBM BlueGene/Q (Green 500): 2 GFlops/W.
-> petascale: HP ProLiant SL390s (Green 500): $<$ 1 GFlops/W.

DARPA target: 20 MW for a 1 EFlop: 50 GFlops/W.

=> a lot of energy-saving engineering has to be developed.

From Challenges to Energy-Aware Services

Enable energy efficiency in the different services.
Power/energy consumption: the main concern whatever the service considered.

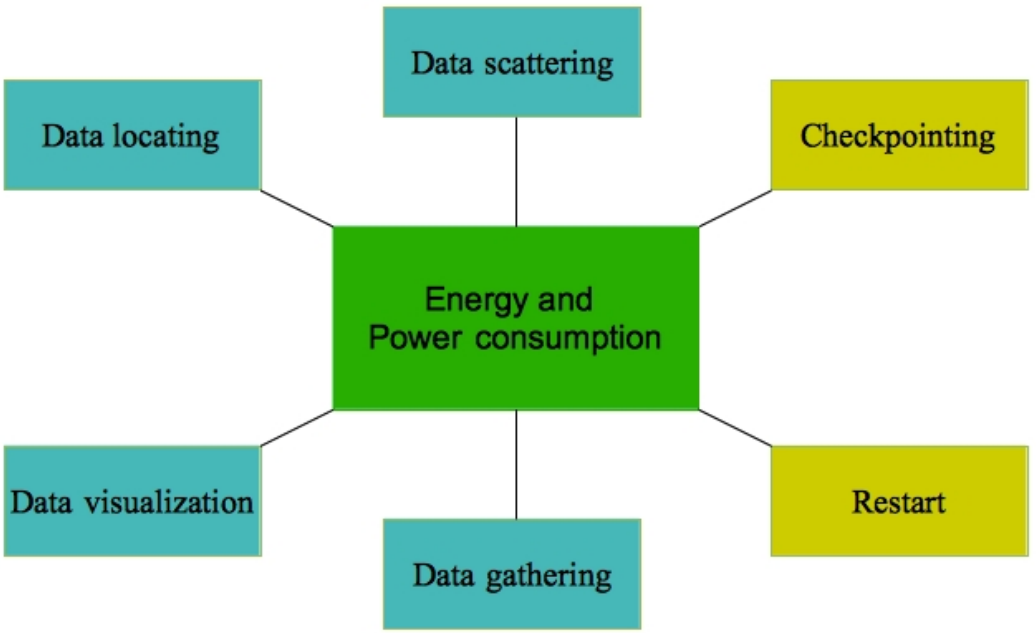

Figure: Services where energy consumption should be considered

Energy-Aware Services

An unified energy-aware framework
==> enabling a coordination of the several energy efficient techniques.

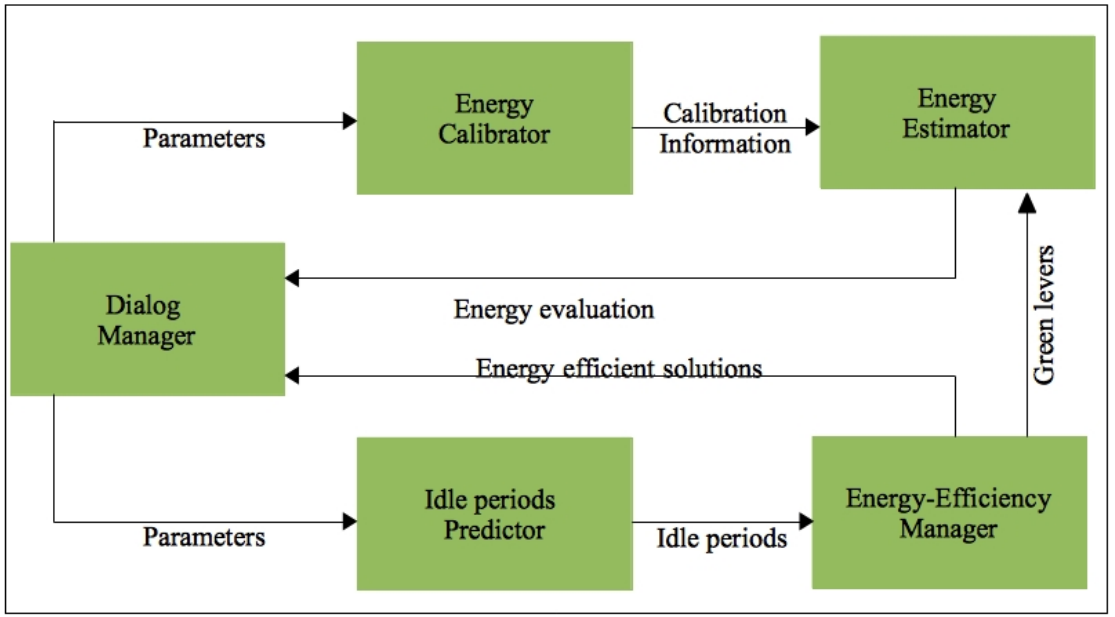

Figure: Framework components

Energy consumption Estimator

For each service, several implementations are possible.

For instance, for fault tolerance: coordinated or uncoordinated checkpointing.
The most energy efficient fault tolerance protocol can be one or the other protocol.

The first step to consume "less": enable the user to choose the less consuming implementation of service.
==> estimate the energy consumption of the different implementations (protocols) of each service.

Energy consumption Estimator

an accurate estimation of the energy consumption of the service implementation depends on several parameters:

- services:
  - fault tolerance: protocol, checkpoint interval, ...
  - data management: scattering/gathering algorithm, ...
- application specifications:
  - Network: size of messages exchanged between processes, ...
  - I/O: volume of data written/read by each process, ...
- hardware:
  - architecture: number of cores per node, ...
  - features: network technologies (Infiniband, ...), ...

Energy consumption Estimator

For instance, energy consumption of data scattering depends on:

- Service: the scattering algorithm
- Application: the number of processes, the volume of data.
- Hardware:
  - the source/destination storage medium.
  - the network interface used (Infiniband, Gigabit Ethernet, ...)
  - the infrastructure architecture: number of cores per node.

In order to compute accurate energy estimations, we must obtain all hardware and application information.

Our approach: calibrate depending on the hardware specifications.

Energy Calibrator

At this end, we developed a set of benchmarks:
==> measure the power consumption and the time execution by varying the different parameters for each implementation of each service.

These measurements serve as a knowledge base.

Besides, a preliminary interaction with the user and the application.
==> information we need as concerns the considered application and the execution context.

Energy-efficiency Manager

Energy consumption of resources is rarely proportional to their usage.
Even if processing nodes are completely idle, they consume significant power.
==> A typical server consumes 175W (idle) and 225W (at its peak usage).

Processes can often be idle or worse actively waiting for a synchronization with other processes
==> Energy is consumed inefficiently.

For example, in case of failure with the uncoordinated checkpoint:

- the crashed processes rollback to their last checkpoints
- the non-crashed processes stay waiting until the restart is done

Energy-efficiency Manager

A very fine-grained cooperation between hardware resources and the application.

We propose to shutdown or slowdown resources during their idle and active waiting periods.
Shutdown approach: dynamically turning off and on resources.
Slowdown approach: dynamically adjusting the performance level.

The shutdown approach is promoted only if the idle period is long enough: due to overheads related to switching off/on. [1].

Shutdown and Slowdown approaches at the resource level: CPU/GPU cores, network interfaces, storage medium.

[1] Orgerie et al, in ICPADS 2008

Idle Periods Predictor

Predict idle and active waiting periods.
==> to know when to apply green leverages (shutdown/slowdown) for each service.

For example, in case of failure, we can anticipate that:
Non-crashed processes will be actively waiting during the restart.

These energy efficient solutions are estimated in terms of energy consumption.

Towards SEASOMES

1. SEASOMES is calibrated: take into account hardware specifications.
2. At each application run, SEASOMES gathers information about the application and the execution context.

==> estimate energy consumption of all the original implementations of the services
==> predict idle and active waiting periods known during the considered services

The dialog manager processes all the information in and out of SEASOMES.

Towards SEASOMES - "consume better"

The energy efficient solutions previously proposed aim at "consuming less" energy

Look for solutions to "consume better"
= consume as much as possible a green energy.
= consume the energy at a lower cost.

In some countries, energy providers offer price per kWh depending on time and day (Example of EDF in France).

Towards SEASOMES - "consume better"

Run applications whenever possible, during off-peak periods so that energy is cheaper and greener.

SEASOMES gathers price information from the energy provider ==> estimate the financial costs of the different services.

The user provides an agenda with different time slots and chooses the most convenient time slot of its agenda by considering together:

- the termination date of the application,
- the energy consumption (in kJ),
- the financial cost.

Towards SEASOMES - "consume better"

Implement a "smart grid" with a double negotiation between energy providers and exascale infrastructures (supercomputer, user, resource manager, ...)

Permanent communication flows:

- Energy provider -> Exascale infrastructure:
  - Green energy => Maximize applications executions.
  - Power capping => Limit application executions up to the available power.
- Exascale infrastructure -> Energy provider:
  - Reduced energy consumption => Disable production of "dirty" energy.
  - Too high peaks consumptions => Provide extra power.

Towards SEASOMES - "External interactions"

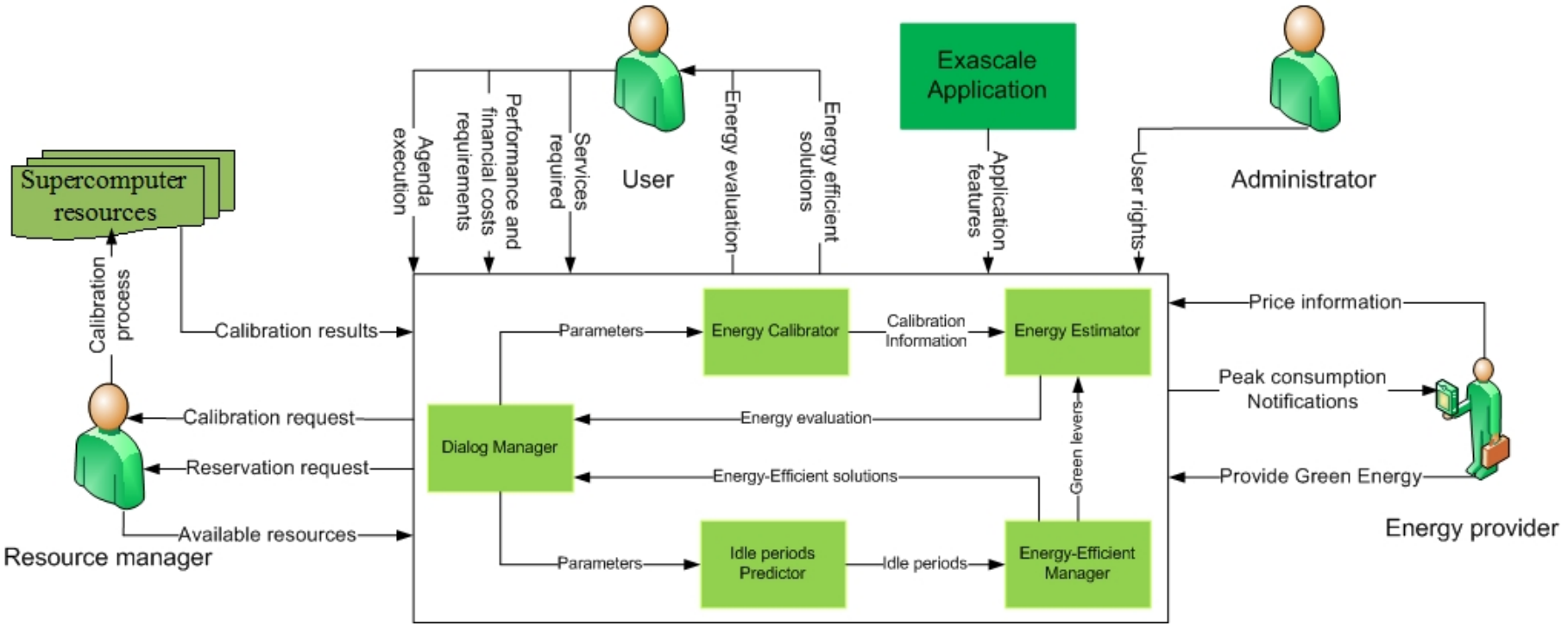

Figure: External interactions involving external actors

Conclusion

A Smart and Energy-Aware Service-Oriented Manager for Extreme-Scale applications: SEASOMES.

The issue of energy efficiency for exascale supercomputers.

Aggregates the various energy-efficient solutions to "consume less" energy and to "consume better".

Both internal (application/hardware) and external (user, administrator, resource manager, ...) interactions.

Future work: Enrich the knowledge base of SEASOMES in order to incorporate a multitude of services.

Conclusion

Thank you for your attention.