Integration of SimGrid in the Datazero project: challenges and solutions

Gwilherm Baudic, Amal Sayah

IRIT, Toulouse
First.last@irit.fr

September 12-13 2018, Troyes
Outline

- Presentation of the Datazero project
- Presentation of SimGrid
- Goals
- Challenges
- Proposed solution
- Demo
- Conclusion
Observation

- Exponential growth of datacenter usage
- Electricity still often obtained from fossil sources (coal)

Context

- DataCenter with several renewable energy sources (wind, solar, fuel cell, battery…)
- Energy production constraints
  - Intermittence
  - Storage
- IT constraints
  - Ensure a quality of service negotiated with the users
The Datazero project (ANR 2015-2019)

- **Goals**
  - Ensure, in a robust and efficient manner, the best possible quality of service for the users
  - Aim towards running solely on renewable power

- **Idea**
  - Implement a negotiation to consider both IT and power constraints
The Datazero project (ANR 2015-2019)

- Several cooperating components/processes
- Communication through an ActiveMQ message bus
- Two implementation scenarios for a component
  - Simulated version, without material constraints, to:
    - Validate the proposed concepts
    - Allow easier observation of the system behavior on time scales which would be unpractical (month, year...)
  - Real implementation:
    - Validate concept feasibility
    - Confront theoretical results to actual observations
    - Using an Openstack architecture
- Possible mix of the 2 scenarios in a single experiment
The Datazero project

Middleware and Datazero components
SimGrid

Open source software supported by INRIA

« SimGrid is a scientific instrument to study the behavior of large-scale distributed systems such as Grids, Clouds, HPC or P2P systems. It can be used to evaluate heuristics, prototype applications or even assess legacy MPI applications. All this as a free software. »
Implementation

- SimGrid
  - Simulation library
  - Infrastructure under study: uses XML file as input
  - Deployment of the application being tested: code or XML

- Reasons for this choice
  - Validated, active project
  - Open source
  - User community
  - Simulation of electrical consumption
Implementation

SimGrid

Usage of SimGrid Java API

• Main MSG simulation Functions
• Process Management Functions
• Host Management Functions
• Task Management Functions
• Mailbox Management Functions
• File Management Functions
• Task Actions
• VMs

More ➔ simgrid.gforge.inria.fr
Goals

✓ Model Datazero concepts
  • Infrastructure
    - Machine
    - Rack
    - DataCenter
  • Activity
    - Job: task to run on servers, either batch or service
    - Phase: part of a Job with fixed resource consumption
    - Flavor: VM characteristics (CPU, RAM…) to run a Job

✓ Interface with the other parts of the middleware
  • Exchange messages through the ActiveMQ bus
    - Receive orders (job arrivals, machine startup…)
    - Send statuses (number of jobs, consumption)
✓ Synchronize SimGrid datacenter events with the other DZ components

• On each Datazero site, activities produce events which need to be globally ordered
  
  → Local scheduling through the internal clock of the underlying system (tasks of a Java VM, processes of a SimGrid simulation...)
  
  → Global scheduling between all Datazero sites, through a client/server application called the Metronome
Challenges

✓ **SimGrid constraints: ActiveMQ messages**

- A synchronous method call blocks the simulation
- Impossible to take into account external events (like message arrivals on the ActiveMQ bus) on an asynchronous fashion
Proposed solution

✓ Projection of the DataZero concepts

<table>
<thead>
<tr>
<th>DataZero</th>
<th>SimGrid</th>
<th>Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Job</td>
<td>Process</td>
<td>DZSimGridProcess</td>
</tr>
<tr>
<td>Phase</td>
<td>Task</td>
<td>DZTask</td>
</tr>
<tr>
<td>Machine</td>
<td>Host</td>
<td>Machine</td>
</tr>
<tr>
<td>DataCenter</td>
<td>main</td>
<td>DataCenter</td>
</tr>
<tr>
<td>Rack</td>
<td>As (zone)</td>
<td>Rack</td>
</tr>
<tr>
<td>Flavor</td>
<td>VM</td>
<td>VmDataZero</td>
</tr>
</tbody>
</table>
Proposed solution

✓ **Implementation without modifying the SimGrid kernel**

- Development of a software overlay to intercept SimGrid "system calls" which have an impact on time

<table>
<thead>
<tr>
<th>SimGrid</th>
<th>Datazero SimGrid</th>
<th>Overload</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process</td>
<td>DZSimGridProcess</td>
<td>Constructor waitFor main exit</td>
</tr>
<tr>
<td>Task</td>
<td>DZTask</td>
<td>execute</td>
</tr>
</tbody>
</table>
✓ **Datacenter in SimGrid**

- Reception of messages to act on the simulation
  - Job arrivals, hardware state change
- Sending of messages on datacenter evolution
  - Job termination, number of jobs in progress
  - Needed, for example for the GUI
  - Periodic sending of observations
- Metronome integration
  - Time advances in simgrid according to messages received from the metronome
  - Send the time corresponding to the next events
  - Using the metronome client and the overloads of Task and Process
Proposed solution

✓ Datacenter in SimGrid (2)

• Addition of a supervision machine to the datacenter
  – Processes received messages
  – Controls the other machines

• New parameters in the XML infrastructure description file
  – Rack consumption
  – Startup/shutdown time of machines with the corresponding consumption

• Application deployment: according to the received messages for job arrivals
Demo: using the metronome

- SimGrid + overload
- Event generator: machine startup, job arrivals
- Java metronome user (non SimGrid)
- ActiveMQ bus
- Metronome
  - The « plain » Java user slows down the simulation
Conclusion

- Need for a datacenter simulator for the Datazero
- Choice: SimGrid
- Issues
  - Management of external events
  - Synchronization with the global time
- Solution
  - Addition of a Java overload to SimGrid to integrate it to the middleware
  - Metronome application to manage the global time
  - No modification of SimGrid C++ kernel
Questions

www.datazero.org