The 1st International Workshop on Autonomic Management of Grid and Cloud Computing
August 5th 2013, Miami, Florida

Clouds and Observatories: How Infrastructure Clouds Can Change the Way We Do Science

Kate Keahey,
keahey@mcs.anl.gov
Argonne National Laboratory, University of Chicago
Cloud Computing: Looking Back

The Need
Environment ownership
On-demand processing

Enabler
Using remote resources

Enabler
Environment ownership
Isolation
Infrastructure Clouds

- environment ownership
- isolation
- as-a-Service
- on-demand & pay-as-you-go
Cloud Computing: Looking Forward

Cloud computing
On-demand, elastic outsourcing

Continuous Data Acquisition
Cheap and reliable “sensing devices”

Event-based data processing
Observatories

8/5/13
www.nimbusproject.org
Ocean Observatory Initiative

- Towards Observatory Science
- Sensor-driven processing
  - An “always-on” service
  - Real-time event-based data stream processing capabilities
  - Highly volatile need for data distribution and processing
- Nimbus team building platform services for integrated, repeatable support for on-demand science
  - High-availability
  - Auto-scaling
- From regional Nimbus clouds to commercial clouds
Building a Plant Observatory

Hyperspectral imaging system mounted on a tower

Spectral Indices*
- NDVI
- SAVI
- EVI
- NDWI

Multi-Temporal Hyperspectral Cubes and Spectral Index Images

Diseases

application images courtesy of Yuki Hamada, ANL

Joint project with Pete Beckman, Nicola Ferrier, Yuki Hamada, Rao Kotamarthi, Rajesh Sankaran and others (ANL)
... and many other projects in flood management, urban metabolism & urban planning, water security, weather forecasting, seismology, etc.
Event-Driven Science

Sensor-based processing

Mobile devices

Support for experiments

Brainstorming sessions

Electronic textbooks

on-demand access to data products and processing
Requirements

• Responsive
  – Support decision-making and infrastructure adaptation
• Adaptive
  – Integrate sensor positioning devices, semi-autonomous vehicles and other actuators
• Dynamic
  – Support dynamic addition and deletion of processing elements
• Scalable and highly available
  – Provide the ability to respond to fluctuating load and failures
• Parsimonious
  – Leverage ongoing computations, in-situ processing, etc.
Data Distribution Network

Data Product 0

A0 ➔ Data Product 0 ➔ A1 ➔ Data Product 1

A2 ➔ Data Product 2

A3 ➔ Data Product 3

A4 ➔ Data Product 4

Subscribe
Architecture

Stream Management

Register stream

Create stream

Subscribe/unsubscribe

Process Management

Register operation

Process Registry

Dispatch Processes

Resource Management

Appliance Registry

Provision Resource
Resource Configuration and Management: the Nimbus Platform
Appliances: Configuration

• Where do VM images come from?
  – BoxGrinder, VMBuilder, rBuilder, veewee, Oz, etc.

• Challenge: interoperability and consistency

• Nimbus Image Portal
  – VM Image creation
  – Generates images for different hypervisors and clouds
  – Based on Packer
Appliances: Contextualization

- Mainstream ctx tools: Chef, Puppet
- Providing abstractions, scalability, repeatability and control
  - StarCluster, Nimbus Context Broker, Nimubs cloudinit.d
- **Nimbus Contextualization**

  - How can I automate multi-dependency deployments?
  - How can I monitor the changes to the environment consistently?
  - How can I create virtual clusters automatically?
  - How can I automate upgrades?
  - How can I securely create a trust layer?

Nimbus Phantom: “The SmartPlayer”

Appliance Gallery

NGVS  STAR  BaBar  PAndAS  ITER

“Player”

General: complex deployments, multi-cloud, autoscaling, high-availability etc.
Multiple models: task farm, master/slave, tightly-coupled virtual clusters
Resource Management Goals

- Scalability and availability: regulates domain properties (compute, storage) using system and application metrics
- Multi-cloud: works with multiple providers
- Brokering: Finding the best resource fit
- Extensible monitoring: VM-based (OpenTSDB, traffic sentinel), provider-based (CloudWatch) and custom
- Flexible, policy-driven: from pre-defined policies to python programs
Phantom: Domain Management

Application-specific qualities: e.g., workload queALiEn, PBS, AMQP, and others

create, manage, destroy

Sensor Input

Domain Manager

Decision Engine

policy

create, manage, destroy

Provisioner

IaaS

CTX

Lifecycle states

Appliances Registry

Paper: “Infrastructure Outsourcing in Multi-Cloud Environment”
Extensible Monitoring

- **Sensor Information:**
  - **External:** lifecycle, contextualization, system sensors (CloudWatch)
  - **Internal:** Contextualization, heartbeat, system sensors (OpenTSDB, traffic sentinel)
  - **Custom:** job queue length, special process monitoring, GPS, and other application-specific sensors
- **Extensible set**
Phantom Interfaces and Clients

- Web application
  - Easy to use – but pre-defined policies

- Scripting
  - REST Phantom API
  - AWS autoscale API (boto-compatible)

- AMQP interface
Adventures in Availability

Mean time between failures

\[
A = \frac{MTBF}{MTBF + MTTR}
\]

Mean time to repair

- **Time to repair (TTR)**
  - Diagnosis
  - **Time to scale (TTS)**
    - PENDING (request)
    - STARTED (deployment)
    - RUNNING (contextualization)

---

**MTBF**

Mean time between failures

**MTTR**

Mean time to repair

**TTS:** preliminary results for 2,000 VMs provisioned on AWS EC2

See also scalability articles at:
www.nimbusproject.org
Managing Processes
Process Management Goals

• Extend process scheduling to integrate an independent resource manager
• Support a variety of existing process schedulers
• Work across cloud and traditional resources
Example: OOI Process Management

Process Instance Registry

Process Definition Registry

OOI Process Dispatcher

Decision Engine

VM Instance

PD agent

enter

lookup

request to run process X

launch
OOI Process Management (cntd)

Domain Manager

Provisioner

IaaS

VM Instance

PD agent

OOI Process Dispatcher

Decision Engine

Process Instance Registry

Process Definition Registry

request to activate process X
Process Management with Swift

Check out the Swift and Torque appliances at www.scienceclouds.org
Using Nimbus Domains

Message Queue
M subtask messages

queue length

Application

Phantom

Cumulus/S3

Infrastructure
Compute Cloud

Define tasks

Gather results

Get task

Results/Checkpoints

www.nimbusproject.org
CloudBurst!

 Phantom

 Workflow

 Process Dispatcher

 Torque Scheduler

 Condor Scheduler

 www.nimbusproject.org
Managing Streams
Stream Processing

• Using AMQP for stream management
• Data processing
  – Control channel vs data channel
• Standard functions, such as archiving and displays become “sink” operations that can be added at any time
Interfaces and Clients

![Easy Stream](image1)

![Register Stream](image2)

![Create Stream](image3)

### "WeatherData Live Stream"

<table>
<thead>
<tr>
<th>Relative_humidity</th>
<th>Temperature10m_faren</th>
<th>Temperature10m_cels</th>
<th>Sigma_theta10m</th>
<th>Windspeed10m_deg</th>
<th>Windspeed10m_dir</th>
<th>Stability</th>
<th>Stability_cname</th>
<th>Net_rad</th>
<th>Solar_rad</th>
<th>Barometric_kpa</th>
<th>Barometric_in</th>
<th>Temperature60m_faren</th>
<th>Temperature60m_cels</th>
<th>Windspeed60m_faren</th>
<th>Windspeed60m_mph</th>
<th>Timestamp</th>
</tr>
</thead>
<tbody>
<tr>
<td>43.</td>
<td>76</td>
<td>24.4</td>
<td>23.2</td>
<td>277</td>
<td>W</td>
<td>-0.13</td>
<td>D</td>
<td>655.</td>
<td>989.</td>
<td>294.</td>
<td>645.</td>
<td>1230 CST on 1 AUG 2013</td>
<td>23.8</td>
<td>10.8</td>
<td>14.4</td>
<td>482.</td>
</tr>
</tbody>
</table>
Status

Proofs-of-concept, prototyping, integration and learning

Integration with existing process managers; enhancements to OOI process manager

Nimbus Platform: basic, production-quality capabilities; refining policies, resource matching, and customization
Open Challenges

• Investigate and understand the interaction pattern
• Data management
  – Data types, transfer/placement management, etc.
• Integration and evolution of process management mechanisms
• Resource management
  – Cloud Resource models
  – SLAs and negotiation
  – Understanding and balancing response time, cost, availability, etc.
  – New repair/recovery strategies
  – Emphasis on networking and storage
• Adaptation: more efficient and reliable adaptation strategies and policies

FutureGrid is a great place to explore them!
www.nimbusproject.org/phantom
Parting Thoughts

Mature Innovation:
The Internet

Maturing Innovation:
The cloud