

# Workflow Project Status Update

---

Luciano Piccoli - Fermilab, IIT

Nov 07 2008

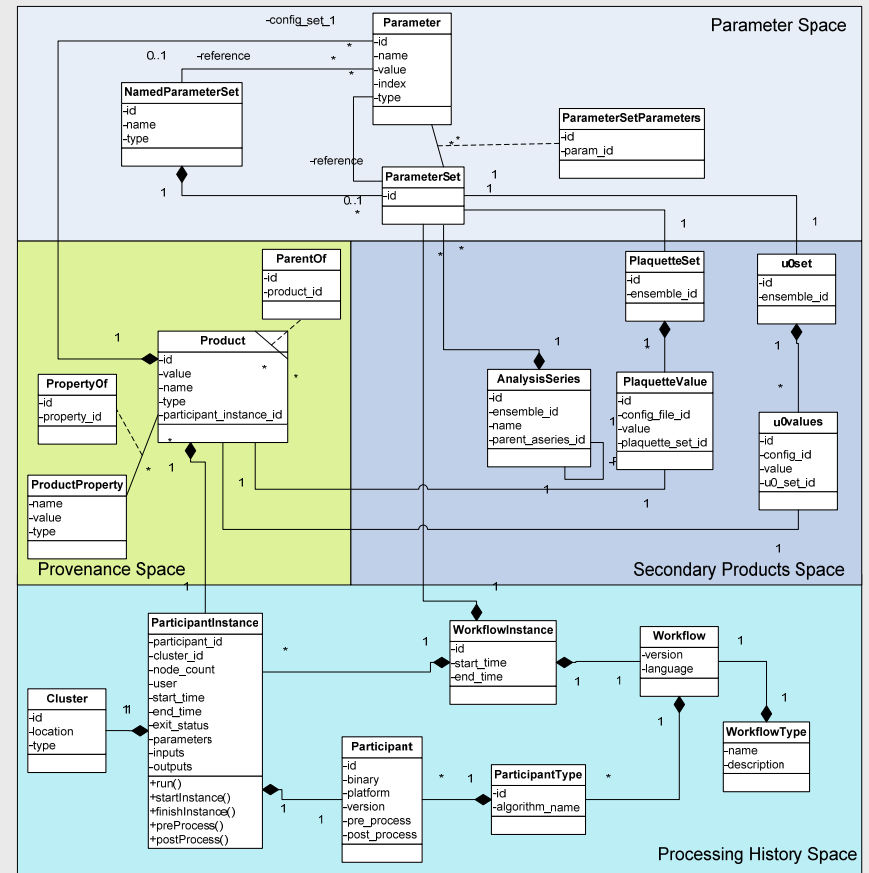
# Reminder

---

- **Workflow:** aims to provide end users with an easier way to orchestrate and describe complex processing of data in a visual form, much like flow charts, but without the need to understand computers or programming.
- **Participant:** workflow task, usually a PBS job. Object that transforms inputs into outputs. Example of participants is: dCache (dccp), PBS (qsub), user applications and shell scripts. All Participants are considered to be atomic operations from the executing workflow's point of view.

# Review

- Change of focus: development of workflow independent front and back-end systems
  - Parameterization
  - Run time history
  - Provenance
  - Secondary data storage
- Ruby on Rails and Ruote BPM workflow engine
  - Configuration generation and 2pt analysis workflows implemented



# Web Interface



## List of Config Parameter Sets

Id	Name(s)	U0 Values	Series	Files	Parameters	Parent Files	Parent Parameters	Workflows	Set Operations	Hash Code
4:	rmd_set	---	tuning	<a href="#">Show</a>	# <a href="#">Show</a>	---	---	<a href="#">Tune</a>	<a href="#">Fork</a> <a href="#">Add Name</a>	62759fa6583db28ad87d70b4fff64246

Based on OpenWFERU

- [Configuration Parameter Sets](#)

## Tuning workflow parameters (set 4)

Node Count:

Algo Name:

Algo Version:

Cluster Name:

# Confgen Workflow (in Ruote)

---

```
class TuneProcessDefinition < OpenWFE::ProcessDefinition
  def make
    cursor do
      get_parameters

      _break :if => "${f:error} != none"

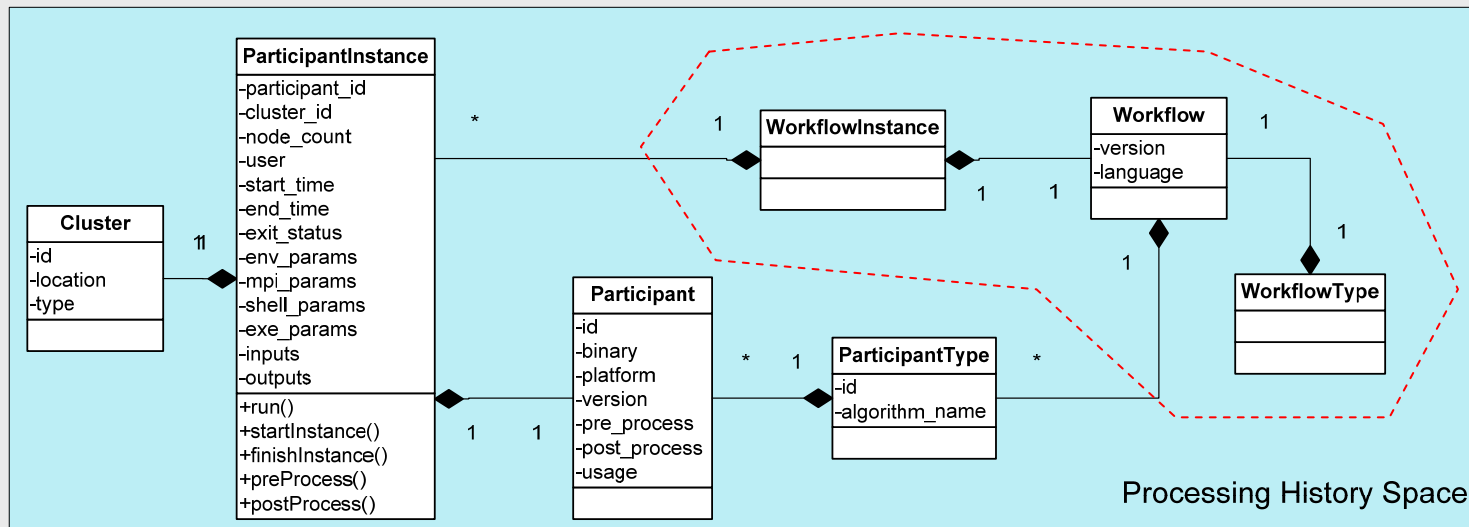
      prepare_tuning

      _break :if => "${f:error} != none"

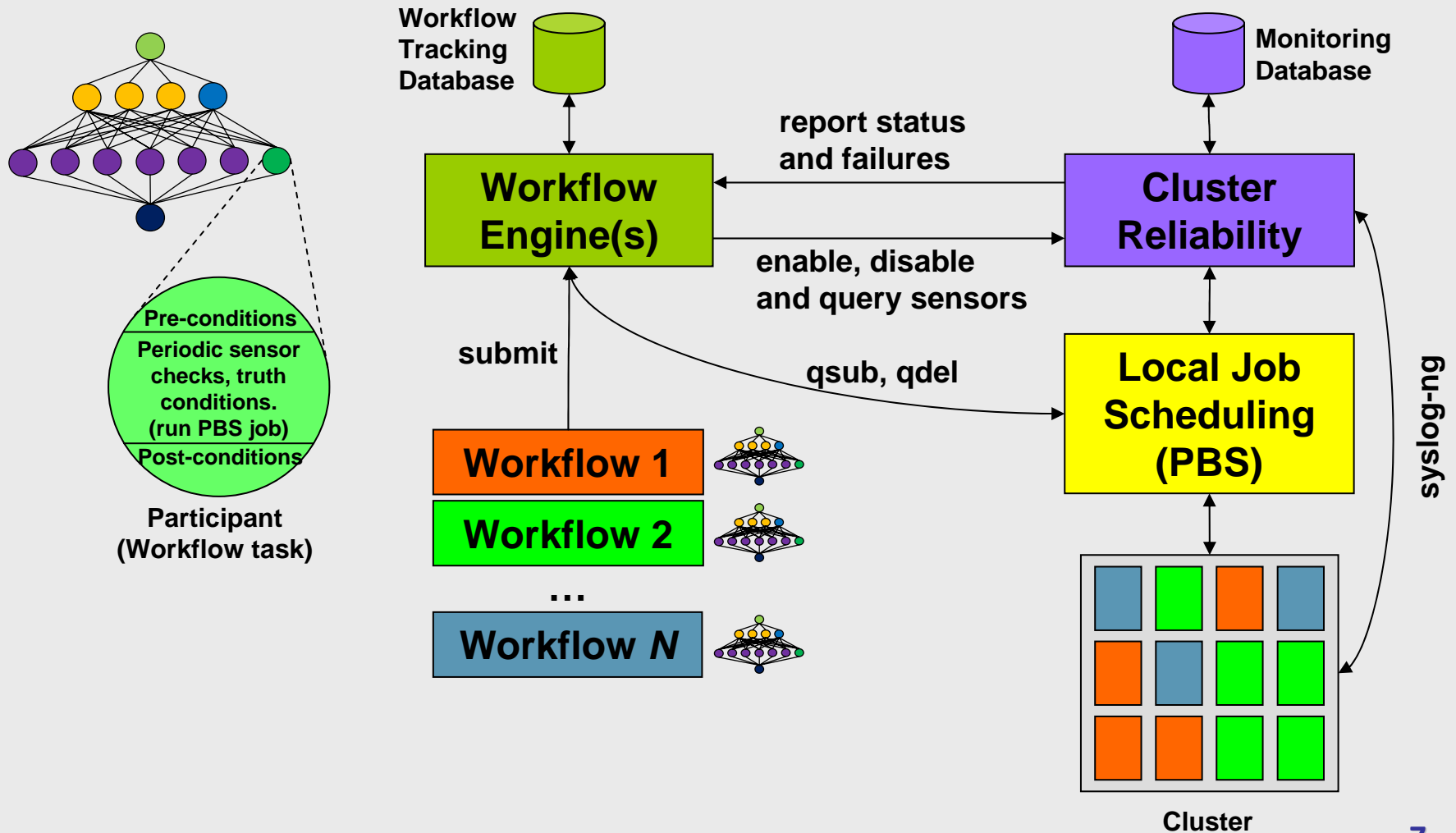
      _loop do
        tune
        check
        _break :if => "${f:done} == true"
        _break :if => "${f:error} != none"
      end
    end
  end
end
```

# Update: Implementation

- Addition of workflow to the data model
  - Allows recovery from failed participant (workflow task)
  - When recovering a workflow existing data products not generated again
  - Currently working on Configuration Generation Workflow



# Update: Integration with Cluster Reliability



# Update: Participation in Conferences

---

- Paper for 'Scientific Workflows and Business Workflow Standards in e-Science (SWBES)' on LQCD workflow requirements and system evaluation.
- SuperComputing08 participation this month
  - Participation in the 3rd Workshop on Workflows in Support of Large-Scale Science (WORKS08)
  - Kepler Tutorial – implementation of confgen/2pt analysis

# Outline for Review

---

# January Review Outline

---

- Motivation for workflow project.
  - What is the LQCD workflow project?
  - What is workflow to us?
    - Parameterized description of physics
    - Grid/SOA vs. Cluster
    - How does it differ from general workflow solutions?
  - Why do we need it?
    - Show how the current process works, show how it would work when using workflows
    - Common definition and storage for physics processes and products (reuse)
    - Increase productivity and accountability
- Description of relevant LQCD workflow requirements, e.g. data handling and fault tolerance.
  - Describe LQCD campaign (2pt analysis and confgen example)
  - Show the parallelism involved by describing the two campaign layers: loop over configurations and processing on single configuration

# January Review Outline

---

- Work that we've done
  - Evaluation of existing workflow systems and collaborations (Swift, Askalon, Ruote (Kepler and Pegasus)).
  - Development of database driven workflow and result tracking system.
    - Workflow independent front and back-end infrastructure
    - Integration with the Cluster Reliability subproject (failure monitoring and diagnostics)
  - Overall workflow/reliability architecture
- Community participation
  - U.Chicago/Swift, Innsbruck/Askalon
  - Lattice 2008
  - E-Science conference 2008
  - SuperComputing 2007, 2008
- Backup slides
  - Real example showing need for fault tolerance: cluster time wasted.