Aviary
A simplified web service interface for Condor

Pete MacKinnon
pmackinn@redhat.com
Drivers

- Condor learning curve
  - ClassAds, Schedds, and Slots - oh my!
  - Terminology doesn't always resemble what an “outsider” might recognize conceptually
  - Arcane details of job scheduling and matching

- Red Hat Grid wants an “ISV-friendly” API
  - Simplicity
  - Ease of access
  - Blueprint laid by previous API development to support RH Grid management console known as Cumin
  - Mr. Oil&Gas: “I have this parallel computation with varying data inputs for our Windows analysis programs. Oh, and let me know when it's done...”
SOAP/HTTP

- XSD & WSDL
  - Schemas are useful in presenting a unified model and communicating design intent
  - SOAP toolkits generally available
  - More on REST later

- But we already have Birdbath!
- Birdbath is relatively fine-grained and assumes the tribal knowledge
  - Begin/end transaction?
  - Add proc to a cluster?
Secret Recipe

- New contrib area
  - `-DWANT_CONTRIB:BOOL=TRUE -DWITH_AVIARY:BOOL=TRUE`
- WS02, Axis2/C for C++ SOAP
  - ASL 2.0
  - Not the darling that gSOAP is
  - Difficult to embed in 3rd-party apps
    - More oriented for httpd deployment
  - Usable codegen for XSD to C++ types
    - Stay away from complex XSD idioms
- DaemonCore and Condor plug-in architecture
  - Barnacle on to schedd
  - DC main to give us integrated log and history reading using Condor internal APIs
  - In both cases, ST Axis listener integrated with DC select()
Model

- Jobs
  - Beyond cluster.proc to be qualified to a pool & scheduler
- Submissions
  - User grouping of job sets that are not related by cluster
  - Open-ended, can update
- Attributes
  - Essentially the ClassAd content of a job
- Catnip to help developers build confidence and proficiency
  - Graduate to ClassAd Ninja over time
- **Not** a SOAP replacement of condor_submit, condor_q, condor_status, etc.
Job Submission and Control

- SubmitJob
  - The usual headline suspects
  - Cmd, Args, Owner, Iwd
  - Requirements – simplified for common selections (e.g., LINUX+X86_64)
  - Extra – optional list of Attributes that support or supplant the other fields

- HoldJob, ReleaseJob, RemoveJob
  - Job id and reason

- SetJobAttribute
Job Queries

- GetJobStatus

- GetJobSummary
  - Cmd, args, queue entry time, last update time, status

- GetJobDetails
  - Return all the attributes from the job's ClassAd

- GetJobData
  - Retrieve the contents of the Error/Log/Output file from a job

- GetSubmissionSummaries
  - Totals of idle, running, held, removed, and completed jobs in each submission group
  - Optionally return job summaries also
SoapUI - submit

```
<soapenv:Envelope xmlns:soapenv="http://schemas.xmlsoap.org/soap/

<soapenv:Header/>

<soapenv:Body>
</soapenv:Body>

<job:SubmitJob allowOverrides="false">
  <cmd>/bin/sleep</cmd>
  <!--Optional:-->
  <args>40</args>
  <!--Optional:-->
  <owner>p.mackinn</owner>
  <!--Optional:-->
  <twd>/tmp</twd>
  <!--Optional:-->
  <submission_name>my_submission</submission_name>
  <!--Zero or more repetitions:--> 
  <requirements>
    <type>OS</type>
    <value>LINUX</value>
  </requirements>
  <!--Zero or more repetitions:--> 
  <extra>
    <name>MYDATA</name>
    <type>STRING</type>
    <value>the data</value>
  </extra>
</job:SubmitJob>

</soapenv:Body>
</soapenv:Envelope>
```
Ruby Savon - submit

# snip out getting client details

xml = Builder::XmlMarkup.new

xml.cmd("/bin/sleep")

xml.args("40")

xml.owner("condor")

xml.iwd("/tmp")

response = client.request :job, "SubmitJob" do

  soap.namespaces["xmlns:job"] = "http://job.aviary.grid.grid.redhat.com"

  soap.body = xml.target!

end
Base Config

WPFCPP_HOME=/var/lib/condor/aviary/axis2.xml

# Aviary Schedd plugin, provides submission and job control endpoint

SCHEDD.PLUGINS = $(SCHEDD.PLUGINS) $(LIB)/plugins/AviaryScheddPlugin-plugin.so

# Port the Aviary Schedd plugin listens on, default 9090
#SCHEDD.HTTP_PORT = 9090

# Aviary query server, provides endpoint for job and submission queries

QUERY_SERVER = $(SBIN)/aviary_query_server

# snip QS details

# Port the QueryServer listens on, default 9091
#QUERY_SERVER.HTTP_PORT = 9091

#QUERY_SERVER.SCHEDD_NAME = 
Future

- Consolidation of QMF and Aviary contribs
- REST
  - POX enabled by Axis2/C
  - JSON support
- Security
  - HTTPS
  - TCP – SSL/TLS
  - WS02,Axis2/C Rampart module (WS-Security)
- Data (and more of it)
  - Aviary currently layered on JobQueueLog and History readers
  - Prefer some kind of DBMS instead
Your turn

Acknowledgments

- Rob Rati – WSO2,Axis2/C packaging
- Tim St. Clair – cmake guidance

Source in 7.6 stream

- WSDL,XSD in etc

We have patched WSO2 2.1-Axis2/C 1.6 to be more flexible with respect to shared library organization

- On track for Fedora 16
- Contact Rob Rati (rrati@redhat.com) for SRPM details
Thank you!

- Demo
- Q&A