Integration Of GenApp with Apache Airavata

Abstract: GenApp is a modular framework for rapid generation of scientific applications. Apache Airavata is a software framework for executing computational jobs on distributed resources. The primary goal of this project is to make Airavata submission from Android and make the new Airavata compatible with the older version thus enabling submission from Qt3,4,5, which will be enhanced to include capabilities currently included only under HTML5. The modules also need to be integrated with Airavata's workflow.

1. Introduction

1.1 GenApp-Introduction

GenApp is a modular framework for rapid generation of scientific applications running on variety of systems including science gateways.

Framework:

This framework builds a user interface for a variety of target environments on a collection of executable modules. The method for execution of the modules is unrestricted by the framework.
Salient Features:

- Allows multiple web-clients to attach to the running job.
- Multiple target applications like HTML5, Qt, Java, etc. can be generated simultaneously for the user, running totally independent of the modules.
- A middleware named Apache Airavata is integrated as an additional execution model within the GenApp framework to support a diversity of queue managed computer resources

1.2 An Induction to Apache Airavata
Apache Airavata is a software framework, dominantly used to build Web-based science gateways and assist to compose, manage, execute, and monitor large scale applications (wrapped as Web based services) and workflows composed of these services.

**Features:**

- Desktop tools and browser-based web interface components for managing applications, workflows and generated data.
- Sophisticated server-side tools for registering and managing scientific applications on computational resources.
- Graphical user interfaces to construct, execute, control, manage and reuse of scientific workflows.
- Interfacing and interoperability with various external data, workflow and provenance management tools.

**Components:**

- **XBaya Workflow Suite** - includes a GUI for workflow composition and monitoring.
- **Airavata Workflow Engine** - the de-facto workflow enacting engine. In future, it’ll be Apache ODE.
- **GFac** - an application wrapper service that can be used to wrap command line-driven science applications
- **WS-Messenger** - a publish-subscribe based message broker implemented on top of Apache Axis2 web services stack.
- **Registry-API** - A thick client registry API for Airavata to put and get documents.

2. Strategic Approach

**2.1 Integration of GenApp with the new Airavata**

- Genapp provides interfaces in various programming languages that need to be connected with Airavata to run various computational jobs.
- The Qt3, Qt4, Qt5 interfaces will be improvised to get Airavata working with the newer version of Airavata and included in the source tree for GenApp. This will be achieved using C++ language interface of Airavata Thrift API.
• This will involve creating calls in the wrapper based on certain variable settings in input Json for targeting execution to Airavata.
• As an additional task, the Qt3, Qt4 and Qt5 versions will be enhanced to include capabilities currently included only under HTML5.

2.2 Workflow Integration

• Each GenApp Module has associated inputs and outputs which can be chained with those of other modules into a workflow to create more complex modules.
• GenApp Modules will be integrated with Airavata which provides interfaces for composition, execution and monitoring of workflows.
• Integration needs to be done with Airavata's Xbaya or Workflow Interpreter to achieve the above. A good design document will be worked upon to achieve this integration.
• Depending upon discussion with the community, a new "XBaya-lite" composition tool (based on Airavata API) could be implemented for composing workflows in GenApp.

2.3 Benefits

• The Qt5: Android framework will be generated for GenApp to enable creation of Android-based applications.
• The integration will be reflected as genapp => languages =>Android.
• The Qt5 Android interface of GenApp generated will be integrated with the new Airavata using Airavata Thrift API.

3. Deliverables

The deliverables to the community from this project include the following:

• Examples of C++ airavata usage (register, etc)
• GenApp Qt3, 4, 5 interface integrated with the new Airavata.
• GenApp Qt5 Android framework generated.
• GenApp Qt5 Android interface integrated with Airavata for executing long computational jobs
• Design Document and/or implementation for Workflow Integration with GenApp
• GenApp Messaging System Integrated with Airavata Messaging System
GenApp Qt3, 4, 5 versions enhanced to include capabilities currently supported under HTML5.

4. Timeline

**Community Bonding Period**

<table>
<thead>
<tr>
<th>April 21 - May 4</th>
<th>2 Weeks</th>
<th>Getting acquainted with GenApp, Airavata codebases and Airavata Thrift API and, understanding the construction of several language frameworks in GenApp.</th>
</tr>
</thead>
<tbody>
<tr>
<td>May 5 – May 11</td>
<td>1 Week</td>
<td>Finalizing the model to generate Qt5 Android framework in GenApp.</td>
</tr>
<tr>
<td>May 12 – May 18</td>
<td>1 Week</td>
<td>Establishing the implementation model to integrate GenApp Android version, and fix the integration of Qt3, 4, 5 versions, with the new Airavata.</td>
</tr>
</tbody>
</table>

**Coding Period**

<table>
<thead>
<tr>
<th>May 19 – June 8</th>
<th>3 Weeks</th>
<th>Improvisation of Qt3, 4, 5 code to enable their integration with the new Airavata.</th>
</tr>
</thead>
<tbody>
<tr>
<td>June 30 – July 27</td>
<td>4 Weeks</td>
<td>Design and implementation of Workflow Integration.</td>
</tr>
<tr>
<td>Date Range</td>
<td>Duration</td>
<td>Task Description</td>
</tr>
<tr>
<td>-------------------</td>
<td>----------</td>
<td>----------------------------------------------------------------------------------</td>
</tr>
<tr>
<td>July 28 – August 10</td>
<td>2 Weeks</td>
<td>Enhancement of Qt3, 4, 5 versions to include capabilities currently supported only under HTML5.</td>
</tr>
<tr>
<td>August 11 – August 18</td>
<td>1 Week</td>
<td>Documentation and Testing.</td>
</tr>
</tbody>
</table>

5. Landmarks

- **Sprint 1**: Getting up to speed with GenApp and Airavata, learning more about their codebase
- **Sprint 2**: Finalizing model for Generation of GenApp Android framework
- **Sprint 3**: Deciding the implementation models to integrate GenApp Android and Qt versions with the new Airavata.
- **Sprint 4**: Integration of GenApp’s Qt3, 4, 5 interface with the new Airavata.
- **Sprint 5**: Generation of GenApp’s Qt5 Android interface.
- **Sprint 6**: Integration of GenApp’s Qt5 Android Interface with the new Airavata.
- **Sprint 7**: Design and implementation for GenApp -Airavata Workflow Integration.
- **Sprint 8**: Enhancement of Qt3, 4, 5 versions.
- **Sprint 9**: Testing, Documentation and Evaluation.

6. Biodata

**Name**: Priyanshu Sekhar Patra

**University**: Indian Institute of Technology (IIT), Kharagpur

**Major**: Pursuing B.Tech in Electrical Engineering (Fourth Year Student)

**Expected Graduation**: May 2015

**Technical Skills**: Android, Java, C, C++, Qt, Python, Javascript, MySQL

**Experience**: I have been involved in the following projects:
• Automatic Identification Of Induction Motor Parameters (*Texas Instruments, Bangalore*)
• Smart Home Management System
• Autonomous Underwater Vehicle
• Image Super Resolution via Sparse Representation

Contact Information:

• **Resume**: Priyanshu Patra - *Curriculum vitae*
• **Email**: pri.patra@gmail.com, pri.patra@ee.iitkgp.ernet.in
• **Skype ID**: priyanshu_psp

7. Availability

**Typical Working Hours**: 9 am - 12 pm, 3 pm - 9 pm IST

**Other Commitments**: None

8. References