User Interface and Analysis Framework

Overview

This section describes the User Interface components and the views that will allow users to interact with the Tupelo beans and launch HPC workflows. If you have not already done so, please review the wiki page describing the Core Framework and Ontology.

Views

Most of the views in the framework will display data stored either by Tupelo or the current PTPFlow Repository. Instead of a single repository view showing everything, there will be multiple views (similar to Medici/Cyberintegrator) that are configured to show a particular type of bean(s) coming from a content provider. The content provider would get the data required from the configured tupelo context. For example, we will need a "Dataset View" that shows all datasets (e.g. input/output datasets) and a way to manipulate them (e.g. add tags, annotations, etc), "Workflow view" that shows all of the workflows that have been ran, "Scenarios View" that shows all scenarios (probably sorted by current user/all users), "Service Registry View" that shows defined RMI service endpoints for launching jobs, and a "Known Hosts View" for showing known hosts that can accept jobs. The last two will be provided by PTPFlow's repository if those PTPFlow objects are not stored in a tupelo context. Each of the views described below that use tupelo beans will extend BardFrameView since the BardFrame will be required to get the information required for each view.

Scenarios View

A primary view provided by the KNSG framework will be the ScenariosView that displays user scenario(s) and all sub-parts (most likely in some kinda of Tree viewer). A scenario is similar to the concept of a project and is simply a way of organizing things that belong together. The scenario is responsible for managing all of the pieces that it contains including input datasets, output datasets and workflows. Users will be able to launch jobs on available HPC machines through an RMI Service (provided by PTPFlow) that use the inputs in their scenario and when a project completes, the outputs should be added back to that scenario, possibly through a thread that is polling the Tupelo server for new data). A user can have multiple scenarios open at once, close scenarios, or even delete scenarios from their scenario view (deleted from the view, but still in the repository) so we'll need to manage which scenarios are in a session and possibly what is their current state (open/closed). The idea of open/closed scenarios is open for debate about whether that is useful to track or if it doesn't matter since no data would be pulled unless a user selected or expanded a scenario. It is anticipated that new applications might extend this view to organize their view differently for their specific domain (e.g. use different icons, possibly organize data into different categories defined by their ontology, etc).

RMI Service Registry View

This view shows all of the machines defined as available to the user for installing the RMI service and PTPFlow plugins required to run HPC jobs and return status information to the client. Below you will see a partial specification for what an RMI Service stored as a tupelo bean might look like. This is not a requirement for version 2.0.

Datasets View

This view will display the datasets that are stored in the tupelo context that the system is connected to. Users should be able to import/export datasets from this view, tag datasets, annotate datasets, etc.

Functional Requirements

1. Import datasets that will be used as input to HPC workflows such as Mesh files, input files (e.g. mach number, poisson ratio, etc)
2. Store output datasets from workflow runs, some workflows will be parameterized and have multiple outputs
3. Export datasets
4. Dataset tagging, Annotation, etc
5. Other functionality?

Workflow view

This view will display the workflows that have been run and the parameters they were ran with (possibly for re-running a workflow).

Functional Requirements

1. Store workflow xml files (Ogrescript) with the parameters used
2. Provide delete, tag, annotate, etc
3. Other functionality?

Overview view

This view will provide general information about what is selected (e.g. a scenario, a dataset, etc). It will display things like who the creator was, date created, etc. If possible, it will provide a preview of what is selected (e.g. a dataset).

Tag view

View for working with tags. It will display all tags associated with the current selection and allow users to manage the tags.

Known Host View

This view contains a list of defined HPC hosts that the user can launch jobs on. This view will provide the user with the ability to change/view/add properties such as environment settings, user information for the host (username, user home, etc), host operating system, node properties, new hosts, etc. These changes should be propogated to the defined RMI services so they can be used immediately.

Analysis Framework

The analysis framework will allow users to register HPC workflows, modify the workflow inputs through a graphical user interface, and execute HPC jobs when all inputs are satisfied. The UI will need information from the metadata to determine if an input is valid for the field (e.g. this field needs a mesh).