Apollo

Project Explorer

Feature list

**confidential**

Revision number: 1

Date:

Name: P. van der Velde

# Goal

The goal for the Apollo Project Explorer is to provide the user with a stand-alone UI for the creation and exploration of Apollo projects.

# Features

For the Apollo Project Explorer a series of user features are suggested. These user features are described below. A few abbreviations are explained below.

**Notation**

TBD To Be Determined

The feature list, sorted by area:

1. **Users**
   1. The Project Explorer is aimed at two different groups of users. These are:
      1. Designers who will use Apollo to design new products and improve existing ones.
      2. Researchers who will use Apollo to investigate the behaviour of different objects in different physical conditions.
2. **Operations**
   1. Global
      1. The user can easily create a new project. The Project Explorer will create the new project and display the data for the newly created project. Note that no geometry can be created in the project explorer.
      2. The user can open an existing project. Upon opening a project the current project will be saved, should the user desire so, and the selected project will be loaded.
      3. The user can save the current project, either with an already provided file name or with a new one. The Project Explorer will ensure that all data is written in a redundant manner so as to always be able to reopen the users file, even if the file is damaged.
      4. The user will be able to close a project without closing the Project Explorer application. When closing a project the user is given the opportunity to save the project if any changes have been applied since the last save action.
      5. The user will be able to import project data from other systems (TBD).
      6. The user will be able to export project data to other systems.
      7. The user will be able to install new components when they are needed. The freshly installed components shall directly be available and no restart of the system will be required to add new components to a project or a data set.
      8. The user will be able to schedule one or more data sets (from the same project) on either the local machine or a remote one. This scheduling will be controlled by the users machine and not by the batch service (even if that’s installed on the users machine).
      9. The user will be able to connect to the batch service for scheduling of multiple data sets of different projects. Data sets can be scheduled on the local machine or on remote machines through the batch service.
      10. The user will be able to create and load scripts for the all actions that run against the user interface. The scripting engine will be able to run all user actions that can be run directly through the UI.
   2. On project
      1. The user will be able to create a new data set which is directly based on the project setup. The new data set will be a direct child of the project and will inherit all project setup. When creating the new data set the user will be able to select the most appropriate data set type, e.g. simulation, theory, experiment.
      2. The user will be able to open the project in the associated CAD application. This allows the user to visualize and compare the data from the different data sets in the project.
      3. The user will be able to add new data types, visualizers or tools to the project. These newly added components will then automatically be included in each of the data sets belonging to the project. Should a data set need more components in order to process the data then this lack will be made clear through non-invasive use of icons and highlighting.
      4. The user will be able to set the different properties for the project. These properties are:
         1. Project goal
         2. Project tolerances
         3. Name
         4. Description
   3. On data set
      1. The user will be able to copy or clone a data set. A copied data set will become a child of the original data set; a cloned data set will become a sibling of the original data set.
      2. The user will be able to lock and unlock data sets. A locked data set cannot be changed by the user or by Apollo. Users will not be able to unlock data sets which are automatically created.
      3. The user will be able to set the different properties for a data set. These properties are:
         1. Name
         2. Description
      4. The user will be able to activate the generators in a data set. Upon activation the generators will process the project and data set information and generate the desired data. During this time the UI will show the progress of the processing in a clear but non-interruptive manner.
      5. The user will be able to add new data objects, visualizers and tools to a data set. The added components are specific to the selected data set and its child data sets.
      6. The user will be able to revert the data in the data set to an earlier save point. This will not revert the child data sets. Child data sets will however be marked as out of date.
      7. The user will be able to delete a data set. If a data set with child data sets is deleted then two options exists:
         1. The child data sets will become children of the parent if the child data sets were created by the user
         2. The child data sets will be deleted if they were created automatically. In this case the user will have the option of keeping the data sets.
      8. The user will be able to export the data into one or more logical disk formats. Data can be either completely exported or only partially exported depending on the export format.
      9. The user will be able to mark restore points for a data set. This allows the data to be restored to this point later on.
3. **User interface**
   1. The user will be able to undo and redo all non-UI actions. In this case UI actions are actions that only work on UI controls (e.g. selecting an item, expanding a tree item etc.).
   2. The project explorer will provide the user with interactive help on several different levels. These levels are:
      1. Statusbar hints: Gives first hint upon mouse over of control
      2. Tool tips: The UI will be able to show tool tips for all different controls
      3. Interactive help: Extended help based on MAML(?)
   3. The project explorer will use visual cues to indicate to the user what is going on. The following states will be shown visually in different places in the UI:
      1. Project
      2. Data sets
         1. Data sets are coloured based on their creation manner. Auto-generated data sets will be more opaque to indicate that the user cannot deal with them directly. Furthermore the context menu for a system generated data set will only show the clone menu item and the properties menu item.
         2. Data sets will have an icon indicating if they are currently processing data, in a paused state or in a stopped state.
         3. Data sets will have a warning icon if the data set is out of date, i.e. when one of the base data sets (parents) has had a change in settings which affect the data set. No warning icon should be shown if the changes in the parent data set do not affect the current data set.
         4. A data set will be highlighted when selected. Furthermore the parents and the path to the project will also be highlighted, although more subtly.
         5. The project explorer will show in what way the data sets will be run, either in batch mode, directly, postponed or other methods. The indication for this status should be available on the data set control.
         6. The data set control will indicate if the data set has been loaded or not.
         7. The data set control will indicate what kind of data set it is, i.e. simulation, experiment or other type.
   4. The project explorer will show the state of the current processing actions. It will show the progress and which actions are currently being performed.

# User Interface

## Screenshots

Main UI:

Project explorer prototype UI

* Project base (geometry etc.) has:
  + Base data (Geometry, Physics models, Boundary conditions, Initial conditions etc.)
  + Visualizations (way data should be shown)
  + Solvers / Equations necessary to translate known data into other known data (e.g. calculation of Mach number from speed and temperature etc.)
* Individual data blocks have:
  + Visualizers necessary to translate between local data format and project data format
  + Additional tools necessary for generation / translation of data
* Expecting to show the user a graph of the project with:
  + The project initial data (geometry, physics models etc.) as the main node.
  + Each child node is a single experiment. Experiments that are directly connected to the parent are user created (?)
  + System created experiments (child experiments) are shown in a different colour (lighter) and cannot be edited (but can be selected)
  + User can drag nodes around (connection stays)
  + User can drag-and-drop nodes onto another node. This changes the links. The dropped node becomes the child of the drop-target node. This only works for user created nodes. Bonds between the child nodes and the dragged node are maintained (important for system generated experiments).
    - Connections cannot be changed. This would imply changing the parent of an experiment which leads to mayhem.
* The graph control should allow
  + Selecting a node. All parent nodes should also be highlighted (but not selected) in weaker colours as they are further away from the selected node. This allows the user to quickly see where the data comes from. The highlighted node will be surrounded by a glow in the system highlight colour.
  + Moving nodes around on the canvas. Nodes should not be able to be moved off the canvas.
  + Allow auto filtering of nodes through the search control. ‘disable’ nodes that do not match with the search criteria. In this case nodes aren’t really disabled but more faded out a bit. Selecting a ‘faded’ node should still be possible
  + There should be a ‘show / hide’ button on / near the parent node to show or hide child nodes. Under normal conditions user created nodes are always displayed while automatically created nodes are hidden.
* When a node is selected the properties for that node are displayed in the properties dialog (use data binding to link).
* When processing an experiment a set of progress controls should be shown. These are:
  + A progress bar
  + A estimated time for the processing + error indication on the time
  + A text block indicating the current (global) action
  + Buttons that allow the user to pause and stop the processing
  + An identifier which identifies the experiment
  + An expander button which can be used to hide / show a block with more advanced details (e.g. which exact action is being processed etc.)
* When the user clicks on a progress block the relevant experiment should be selected. Ditto if the user selects an experiment that is being processed then the progress block should be selected.
* If an experiment has child experiments which have to be processed when processing the parent then the progress block should show the progress of the parent experiment. The progress of the child experiments should be shown in progress blocks which are children of the parent progress block. It should be possible to hide these.
* Parent progress blocks should also indicate if the parent experiment is in a waiting state or not.
* Remote running experiments should also show a progress control + icon. This indicates that the experiment is being run remotely.
* If the user opens up a project for which one or more experiments are being run remotely then the UI should directly reflect that (minus any network issues).
* The UI should be able to display an error list which indicates to the user where errors have occurred. Note that errors for a data set / project should be shown on the visual element for these items.

Questions:

* What should nodes display? Do we allow a node to expand or do we show stuff in the properties section of the application?
* How do we show tools / visualizers / data?

Non-features:

* No displaying of geometry
* No setting variables / properties on geometry