Apollo

Data storage specification

confidential

Revision number: 1

Date:

Name: P. van der Velde

# Introduction

There are two main data storage parts, the in-memory storage and the file storage. Both have different goals

# In-memory data storage

Goal:

To keep the data that is currently in use by Apollo. Should allow users to manipulate that data without corrupting it or presenting an inconsistent state. Should also allow users to get back to an earlier state if this is desired.

DATA STORAGE

* Make sure data is always consistent. Can use transactions (commands) for this.
* It should be possible to write time based data as a series (good for moving meshes, unsteady sims etc.)
* It should be possible to write data as a series for undo-redo. This may mean that time-based data gets more complex? Should we store version numbers for each ‘change’?
* Data also stores component info etc.
* Allow data to be real data (data that can be used by components other than the generator) or ghost data (data that hasn’t really been committed, only usable by the generator and a visualizer). This is useful for suggested changes in e.g. the mesh generators etc.
* Data must have a state. Which can be:
  + Up to date
  + Invalidated (give info about why?)
  + Ghost (?)

UNDO / REDO

SNAPSHOTS

* Allow snapshots of the complete system. These should be able to be streamed to disk or other storage and contain a complete simulation / project. Users should be able to use these snapshots directly
* Allow the system to make snapshots automatically and return to a snapshot if things fail. Maintain a graph of snapshots (for undo purposes)
* The snapshot engine should be able to reset the simulation / project back to a specific snapshot but should ask components if any non-persistent data should be passed on to itself after restoring the snapshot (sort of information from the future)
* All data components that are persistent should support snapshotting + undo + transactions. These can all be defined through commands etc.
* Snapshots should be version resistant. Newer systems should be able to load older snapshots
* A snapshot should be read-only. If the user wants to make changes then they’ll have to save the project / simulation separately.
* Snapshots should be robust. Corruption in the snapshot should be repairable
* Allow a snapshot to be copied to the head of the snapshot graph (for data resets)

# File data storage

* Store CRC’s etc. to make the file robust against corruption.