MPI ~~Tool~~ Interfaces

Outbrief

August 2015

Agenda Before Granlibakken

- Goal is to redesign the MPI Profiling Interface while keeping all existing functionality
  - No longer rely on weak symbols
  - Support multiple tools and allow composability
  - Clean Fortran support without writing tools in Fortran
- Main idea: Callback interface
  - Tool startup allows multiple tools to register themselves
    - Tool initiates it's own registration
    - Start-up protocol/handshake during MPI_Init
  - Creating of a tool DAG (stackable tools)
  - Maintain the wrapping idea
  - Configurable "out-calls" instead of fixed PMPI calls

Agenda After Granlibakken

- Discussion of role of tools led to initial misunderstandings
  - Tools as profilers (>>Profiling<< Interface)
  - Tools seen more general (debuggers, correctness, …)
  - Applications extensions (e.g., fault tolerance)
- Goal is actually a more general Extensible MPI Interface
  - More than tools -> plugins
  - "Justifies" extra complexity
    - Multiple tools are necessary to enable tools plus application extensions
    - Useful to express dependencies
  - New PMPI interfaces is a "side product" at the end
  - Opens the door to many more use cases
- Useful for more than just MPI / could be used for any API

Open Issues / Wishes / Requests

- At least limited ABI compatibility
  - Make core interface compatible
  - Allow for bundles tools for multiple MPIs as one library
- Tool/Plugin configuration
  - How to express dependencies of plugins?
  - How to combine system/plugin/user configurations?
  - Priorities?
- Open Issues
  - How does this relate to spawn?
  - How to handle proper finalize of multiple plugins?
  - How to handle simple cases with a few MPI routines only?

https://svn.mpi-forum.org/trac/mpi-forum-web/wiki/MPI3Tools