LA-UR-12-22403
Approved for public release; distribution is unlimited.

Title: Integration of Hydra-TH in VERA (L2 Milestone THM.CFD.P5.01)

Author(s): Christon, Mark A.
Baksoi, Jozsef
Barnett, Nathan
Francois, Marianne M.
Lowrie, Robert B.
Sankaran, Ramanan

Intended for: Report

Integration of Hydra-TH in VERA (L2 Milestone THM.CFD.P5.01)

Jozsef Bakosi, Nathan Barnett, Mark A. Christon,
Marianne M. Francois, and Robert B. Lowrie
Computational Physics Group (CCS-2)
Computer, Computational and Statistical Sciences Division
Los Alamos National Laboratory
Los Alamos, NM 87544

Ramanan Sankaran
Scientific Computing Group
Center for Computational Sciences
Oak Ridge National Laboratory
Oak Ridge, TN 37831

With contributions from Roger Pawlowski, Oak Ridge National Laboratory

1 Executive Summary

This report describes the work carried out for completion of the Thermal Hydraulics Methods (THM) Level 2 Milestone THM.CFD.P5.01 for the Consortium for Advanced Simulation of Light Water Reactors (CASL). This milestone focused primarily on the initial integration of Hydra-TH in VERA. The primary objective for this milestone was the integration of Hydra-TH as a standalone executable in VERA.

A series of code extensions/modifications have been made to Hydra-TH to facilitate integration of Hydra-TH in VERA and to permit future tighter integration and physics coupling. A total of 61 serial and 64 parallel regression tests have been supplied with Hydra-TH. These tests are are being executed in the TriBITS environment. Once the VERA team enables the full suite of tests, the results can be posted to the VERA CDash site. Future work will consider the use of the LIME 2.0 interface for tighter integration in VERA with additional efforts focused on multiphysics coupling with radiation transport, fuel performance, and solid/structural mechanics.

2 Introduction

This milestone was comprised of 9 sub-tasks with the primary objective being the integration of Hydra-TH as a “standalone” executable in VERA. The 9 sub-tasks for this milestone included:

1. Wrap all source code in a unique ”Hydra” namespace,
2. Provide a switchable command-line parser for use with VERA’s “main”,
3. Provide a Hydra driver interface that accepts an MPI communicator, file names, etc., in order to satisfy LIME requirements,

4. Integrate a "backup" method in the virtual physics solve to permit repeating a time-step or iteration,

5. Define all output streams,

6. Establish ORNL repository of code base,

7. Decision point on September 30, 2012 release of Hydra-TH in VERA,

8. Integration of Hydra-TH with TriBITS as a standalone code in VERA,

9. Document all results.

In §3, the primary sub-tasks are outlined with a summary of the current status of each sub-task. In §4 a brief summary of this milestone are presented and future directions for this work are outlined.

3 Milestone Accomplishments

The delivery of Hydra-TH for VERA has been a multi-institutional effort relying heavily on the expertise of technical staff at both LANL and ORNL. The integration effort followed the requirements for code integration into VERA defined by Roger Pawlowski at the December CASL THM Workshop [2].

3.1 Integration of HYDRA-TH in VERA

The primary focus for this effort was to integrate Hydra-TH as a stand-alone executable in VERA using TriBITS. Therefore, some of the tasks presented below were primarily pro-active in preparation for future tighter integration of Hydra-TH in VERA.

3.1.1 Hydra Namespace

As a prerequisite, all of the hydra classes were wrapped in a unique C++ namespace, `Hydra`. This permits all of the functionality presented by Hydra-TH to be integrated into VERA without variable, class or function name collision with other libraries in VERA.

3.1.2 Hydra Driver

The Hydra Driver is a class that permits Hydra-TH to be instanced as a library with an interface that is easily extensible for integration into VERA. This interface currently provides basic functionality for initialization, setup (i.e., creation of Hydra objects, I/O, run-time parallel load-balancing, etc.), and physics solution, timing, and finalization. This interface is easily integrated with exception handling to permit robust integration into VERA. This interface permits switchable command-line parsing to allow Hydra-TH to be built and executed as a stand-alone code with its own command line parsing, and is easily extended to accept a list of objects describing the relevant keyword, mesh input files, output files, etc.

3.1.3 Backup in `Physics::solve()`

This functionality is intended to permit Hydra's solution algorithms to backup in time if an overall multiphysics time-step in VERA fails for some reason. The `solve()` method is integral in Hydra's Driver interface. Extensions to this interface specific to LIME 2.0 can be easily implemented into the existing Driver class that is presented to VERA for integration.

At this time, the "backup" functionality can be easily handled using Hydra's existing data registration interface that is integral to the basic data container in Hydra. This data container is used for the registration of the solution variables with attributes that identify specific variables for output, restart, etc. This interface in the the data container will be used to identify all registered variables that must be cached during a time-step in case of a potential "backup event". In the case of a backup, all registered variables that have been identified for backup can simply be restored to their start of time-step state.

The design for the "backup" functionality is complete. However, due to uncertainties in the LIME 2.0 interface and the requirement to deliver Hydra-TH as a standalone code for this milestone, the final interface has not been coded. When the LIME 2.0 interface is fully defined, and Hydra-TH is more tightly integrated into VERA, the final driver interface will be coded with extensions to the existing design as required by VERA.

3.1.4 Define Output Streams

Hydra manages the streams for all of its primary input/output mechanisms through its I/O class. User and diagnostic output is typically handled separately. A preliminary design to permit specification of the streams used for user/diagnostic output by VERA is in place. However, due to uncertainties in the LIME 2.0 interface at the time of this writing, and the fact that Hydra-TH is currently being integrated as a stand-alone application, the final interface has not been coded. When the LIME 2.0 interface is fully defined, and Hydra-TH is more tightly integrated into VERA, the final driver interface will be coded as required.

3.1.5 ORNL Hydra-TH Repository

This task was performed in close collaboration with Ramanan Sankaran at ORNL. In order to facilitate development by multiple developers from different organizations, the following requirements were defined for the Hydra-TH repository at ORNL.

- One repository (Hydra-TH) holds only a subset of a master repository (Hydra)
- A number of developers may clone and contribute to either repository
- Some developers are only authorized to access the subset repository
- The subset should never contain the files (or their history) that are only in the master repository
- Commits recorded in either repository should be propagated to the other via a filter:
  - Changes coming in the master propagate to the subset only if they contribute to the subset
  - Commits coming in the subset freely propagate to the master
- Commit propagation should be automatic and done by the repositories, i.e. without external scripts or manual editing the repository meta-data, etc.

The master repository (hydra) is hosted by SVN at LANL and sits on a "yellow" network that is not generally accessible from the public internet. The subset repository (hydra-th) is a bare git repository. Communication between the two is handled via a third (git) repository (hydra-th.trans) hosted at LANL. The only task of hydra-th.trans is to propagate changes to and from hydra and hydra-th. The filter in hydra-th.trans is implemented via git-svn's ignore-paths option, which facilitates a "filtered fetch" (including fetches as part of dcommit, rebase, etc). Figure 1 shows the relationship between the hydra SVN repository at LANL and the hydra-th git repository at ORNL.

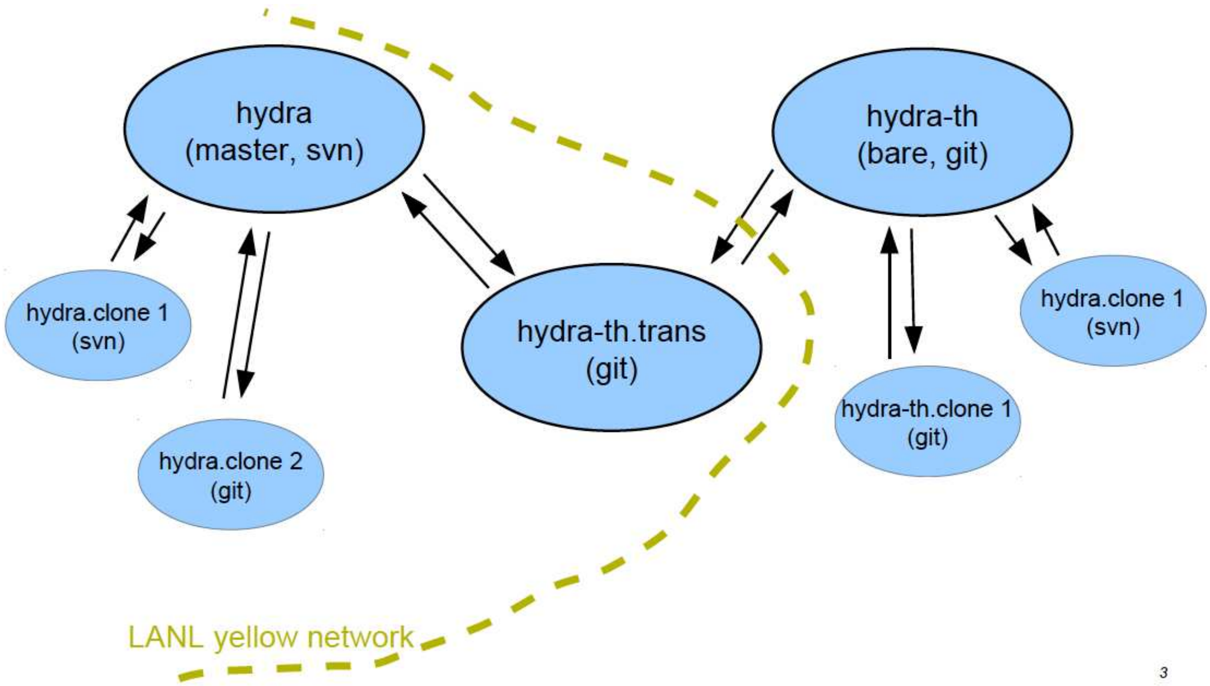


Figure 1: Hydra-TH repository configuration.

We note here that the hydra-th (git) repository also includes a relatively complete regression suite complete with input meshes, control files and baseline results. A test suite containing 61 serial tests and 64 parallel tests has been included. A python test driver is included as well, and at this time, all of the tests pass on the so-called fissile-4 machines at ORNL.

3.1.6 Decision Point

At the time of the 2012 CASL Virtual Roundtable, the probability that Hydra-TH will be included in the September release of VERA remains "medium". At the time of this writing, the primary requirements for including Hydra-TH in the VERA release have been met.

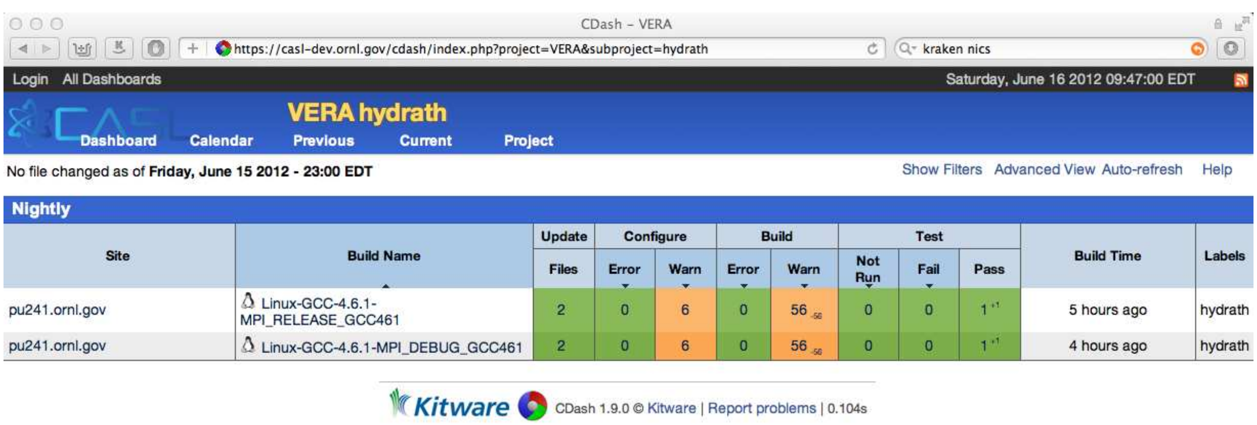
3.1.7 Integration with TriBITS

Hydra uses CMake [1] for builds across multiple platforms. In the Hydra-TH git repository, suite of third-party libraries that include Trilinos, ParMetis, Exodus-II, PETSc, etc. are included. For the stand-alone version of Hydra-TH in VERA, these libraries are currently built independent of the VERA third-party libraries.

Under TriBITS, an auxiliary set of CMakeLists files have been developed and co-exist with Hydra's default CMakeLists files. This permits Hydra-TH builds under TriBITS at ORNL, while preserving the stand-alone build procedure used for Hydra.

A complete suite of serial and parallel regression tests are included with the Hydra-TH git repository. At this time, the full suite of serial and parallel test suites are running under the TriBITS system. Figure 2 a) shows the build and test results for Hydra-TH for the TriBITS continous integration (third row), and Figure 2 b) shows the nightly build and test results for Hydra-TH. Once the VERA team enables the full suite of Hydra-TH tests, they can be posted to the VERA CDash site.

a)



b)

Figure 2: Build & test results for Hydra-TH for a) continuous integration (third row), and b) nightly integration.

4 Summary and Future Directions

A series of tasks have been carried out as defined by the THM L2 milestone THM.CFD.P5.01 for the integration of Hydra-TH into VERA as a standalone executable. Due to variability in levels of network security, a series of three code repositories have been setup to permit distributed development of Hydra-TH with the ability to push changes in a bi-directional manner. This permits code changes at LANL to be easily migrated to the ORNL repository for Hydra-TH, and conversely, extensions to Hydra-TH to be pushed from ORNL to LANL while preserving the overall history of changes to the code.

A number of code extensions/modification have been made to Hydra-TH to permit the current stand-alone integration into VERA, and to facilitate future tighter integration into VERA with coupling to other physics, e.g., radiation transport, fuel performance, solid/structural mechanics, etc. It is anticipated that additional, but minor, interface changes will be required in the Hydra-TH Driver to accommodate the LIME 2.0 interfaces. Because the LIME 2.0 interface is not available yet, no attempt was made to achieve a tighter integration into VERA.

A total of 61 serial and 64 parallel regression tests have been supplied with Hydra-TH. These tests are being executed in the TriBITS environment. Once the VERA team enables one or more of these tests, the results can be posted to the VERA CDash site.

As Hydra-TH development moves forward we anticipate a series of additional development steps that include

- Integration into VERA using the LIME 2.0 interface
- Coupling to radiation transport and fuel performance codes
- Integration of the existing Conjugate Heat Transfer (CHT) and fluid-solid interaction (FSI) interfaces in Hydra-TH with LIME 2.0 for performing CHT/FSI calculations with third-party heat-conduction and solid/structural codes

ACKNOWLEDGMENTS

The authors would like to thank Roger Pawlowski for his helpful input, sample CMake files and discussions regarding integration of Hydra-TH for VRI.

References

[1] K. Martin and B. Hoffman, *Mastering CMake – a cross-platform build system*, Kitware, Inc., 2007.

[2] R. Pawlowski, *How will VERA-CFD couple into VERA?*, in CASL THM Workshop, December 2011.