IRIS-HEP Fellowship Proposal
Improve Cling's packaging system: Cling Packaging Tool

Project Area: Compiler Infrastructure

Surya Somayyajula

somayyajula@wisc.edu | University of Wisconsin-Madison

Mentor: Vassil Vassilev

Background

Cling is an interactive C++ interpreter/compiler that utilizes the REPL (read-evaluate-print-loop) paradigm for fast development and testing as well as immediate feedback and runtime-generated code. Cling is built on top of Clang and LLVM compiler infrastructure, and as a result, has many useful features such as expressive compiler diagnostics, and in addition, also has its own command line prompt and uses a JIT (just-in-time) compiler. Not only is Cling useful for rapid development, the functionality of an application can be improved by embedding Cling.

One of the many useful tools included in the Cling interpreter is the Cling Packaging Tool (CPT), which is a command line utility that can easily build Cling from source and generate installer bundles for a variety of platforms, including Ubuntu and Debian-based platforms, Windows, distributions based on Red Hat Linux, Mac OS X, and any Unix-like platform. While the CPT is an incredibly useful and flexible tool, there are several improvements that can be made to make the user's experience with the CPT even more seamless.

Proposed Project

To advance the CPT, this project proposes several improvements to the CPT and documentation that will enhance the user experience. They would be implemented as follows:

1. *Fixing platform issues*
    a. Fixing builds using LLVM prebuilt binary packages on Linux and MacOS and adding that to continuous integration
    b. Fixing builds using LLVM installed from package manager on Linux and MacOS and adding that to continuous integration
    c. Enable builds using pre-built binaries on Windows and adding that to continuous integration
    d. Using Homebrew as the default package manager on MacOS instead of macports
    e. Fixing Debian packaging creation

2. *Rewriting parts of CPT*
    a. Using a different program execution starting point
    b. Getting rid of the mutation of global variables
    c. Minimizing the use of the subprocess module
    d. Making CPT flake8 (code linter) compliant (allows for flexible error/violation codes)
    e. Rewriting the argument parser
        i. The CPT is both an installer and packager, so the feature to build Cling without having to package it will be added
        ii. Enabling dependent arguments, so it should only be possible to pass some arguments only if another specific argument is passed
        iii. Fixing the naming of the arguments, there should be an established uniformity
    f. Will comment out broken features so users do not run into errors
    g. Add feature to specify the number of cores of CPU to use when building Cling
    h. Will add feature to check if installed dependencies are always detected by the check requirements flag (regardless of the package manager used) and verify the list of required dependencies

3. *Rewriting documentation*
    a. The Overriding Default Variables section on the README page and tools/packaging is outdated, so updating the instructions with the correct flags to use would help the user
    b. Adding documentation for new features added during this project

4. *Fixing miscellaneous issues*
    a. Fixing the verbose flag in the CMake log to ensure for printing out verbose output
    b. Adding cmakeDir to CPT to fix compilation of CPT

Timeline

Week	Goal
1	1. Reading Cling ROOT documentation and getting to know the developer community through the GitHub bug tracker and cling-dev mailing list 2. Project setup on Ubuntu terminal environment with Windows Subsystem for Linux 3. Using a build of Cling and familiarizing myself with the codebase and usage for CPT and Cling according to the documentation
2	1. Discuss with the community and my mentor the plan for rehauling CPT, by sending an RFC email to cling-dev mailing lists discussing the plan 2. Starting to fix *miscellaneous issues* and other small issues such as issue no. #426 to get an idea of how the workflow will follow with subsequent tasks
3	1. Finish fixing the *miscellaneous issues* such as the verbose flag issue listed in the meta issue list #406 2. Replicating environment and commands in which *miscellaneous issues* appeared in the first place, and seeing if the build passes, will ensure that the issues have been fixed 3. Start *rewriting the CPT* (part a will be completed) (part a listed in meta issue list #406) 4. Document the progress made *rewriting the CPT* and fixing the *miscellaneous issues*, and update the documentation accordingly 5. Running and debugging a build of Cling to check that the CPT is executing at a different starting point

4	1. Finish part b of *rewriting the CPT* (part b listed in meta issue list #406) 2. Complete *rewriting documentation* part a (issue no. 372) 3. Document the progress made *rewriting the CPT*, and update the documentation accordingly 4. Debugging a build of Cling and run CPT to check that the global variables are not mutated
5	1. Finish part c of *rewriting the CPT* (part c listed in meta issue list #406) 2. Document the progress made *rewriting the CPT*, and update the documentation accordingly 3. Debugging a build of Cling to check that the CPT still works the same after the removal of subprocesses
6	1. Finish part d of *rewriting the CPT* (part d listed in meta issue list #406) 2. Finish part a of *fixing platform issues* (part a listed in meta issue list #406) 3. Document the progress made *rewriting the CPT* and *fixing platform issues*, and update the documentation accordingly 4. Running flake8 on CPT to see if there are any violations 5. Debugging a build of Cling that uses an LLVM pre-built tarball and running CPT on the Ubuntu terminal environment with Windows Subsystem for Linux and a macOS virtual machine and see if everything is functioning according to specification
7	1. Finish part e of *rewriting the CPT* (part e listed in meta issue list #406) 2. Finish part b of *fixing platform issues* (part b listed in meta issue list #406) 3. Document the progress made *rewriting the CPT* and *fixing platform issues*, and update the documentation

	accordingly 4. Debugging a build of Cling and running CPT to see if the CPT commands still function 5. Debugging a build of Cling that is using LLVM installed from a package manager and running CPT on on the Ubuntu terminal environment with Windows Subsystem for Linux and macOS virtual machine and see if everything is functioning according to specification
8	1. Finish part f of *rewriting the CPT* (part f listed in meta issue list #406) 2. Finish part c of *fixing platform issues* (part c listed in meta issue list #406) 3. Document the progress made *rewriting the CPT* and *fixing platform issues*, and update the documentation accordingly 4. Debugging a build of Cling and running CPT to see if any broken features are still showing up 5. Debugging a build of Cling and running CPT and see if everything is functioning according to specification on Windows
9	1. Finish part g of *rewriting the CPT* (issue no. 391) 2. Finish part d of *fixing platform issues* (issue no. 404) 3. Document the progress made *rewriting the CPT* and *fixing platform issues*, and update the documentation accordingly 4. Debugging a build of Cling and running CPT and check if the specified cores feature is running correctly according to specification 5. Debugging a build of Cling and running CPT to see if the Homebrew package manager is being used
10	1. Finish part h of *rewriting the CPT*

	(issue no. 404) 2. Document the progress made *rewriting the CPT*, and update the documentation accordingly 3. Debugging a build of Cling and running CPT and checking if the check requirements flag correctly checks for installed dependencies and verify the list of required dependencies
11	1. Finish part e of *fixing platform issues* (issue no. 307) 2. Document the progress made *fixing platform issues*, and update the documentation accordingly 3. Debugging a build of Cling and running CPT on a Debian virtual machine and checking if package creation is working
12	1. Implementing any more fixes if there is time 2. Finish updating documentation

Time Zone

Eastern Standard Time - UTC-05:00

Commitments

I don't have any other commitments for this summer, so I would be able to work full-time, 40 hours a week for the full summer (12 weeks, late May-late August). My hours are flexible, and I can adjust to times that would work for my mentor.

References

1. https://github.com/root-project/cling/issues?q=is%3Aissue+cpt+is%3Aopen
2. https://github.com/root-project/cling
3. https://rawcdn.githack.com/root-project/cling/d59d27ad61f2f3a78cd46e652cd9fb8adb893565/www/index.html
4. https://root.cern/cling/