COMPILER RESEARCH TEAM
Princeton University

Date
13 November 2025

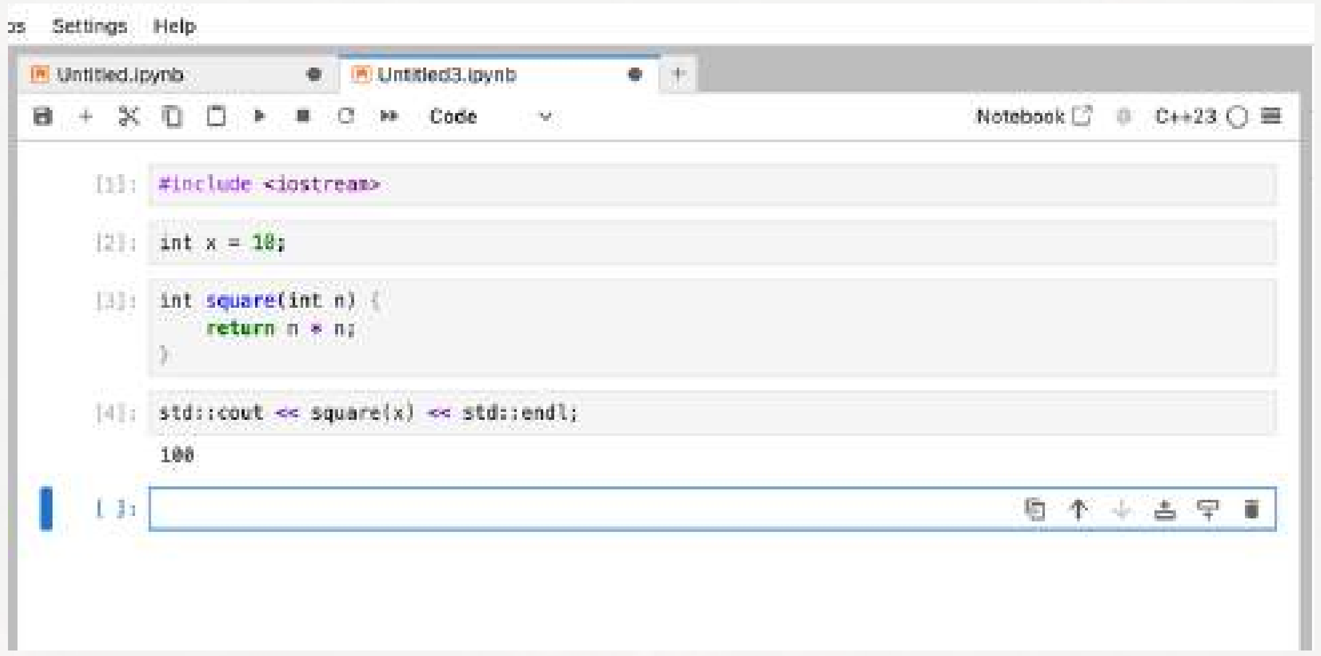
Implementing Debugging Support for xeus-cpp

GOOGLE SUMMER OF CODE 2025- FINAL PRESENTATION

Mentors: Anutosh Bhat, Vipul Cariappa, Aaron Jomy, Vassil Vassilev

Mentee: Abhinav Kumar

About xeus-cpp



```
[1]: #include <iostream>

[2]: int x = 10;

[3]: int square(int n) {
         return n * n;
     }

[4]: std::cout << square(x) << std::endl;
     100

[ ]:
```

1. Xeus-Cpp is a Jupyter kernel that enables interactive C++ programming within the Jupyter environment.

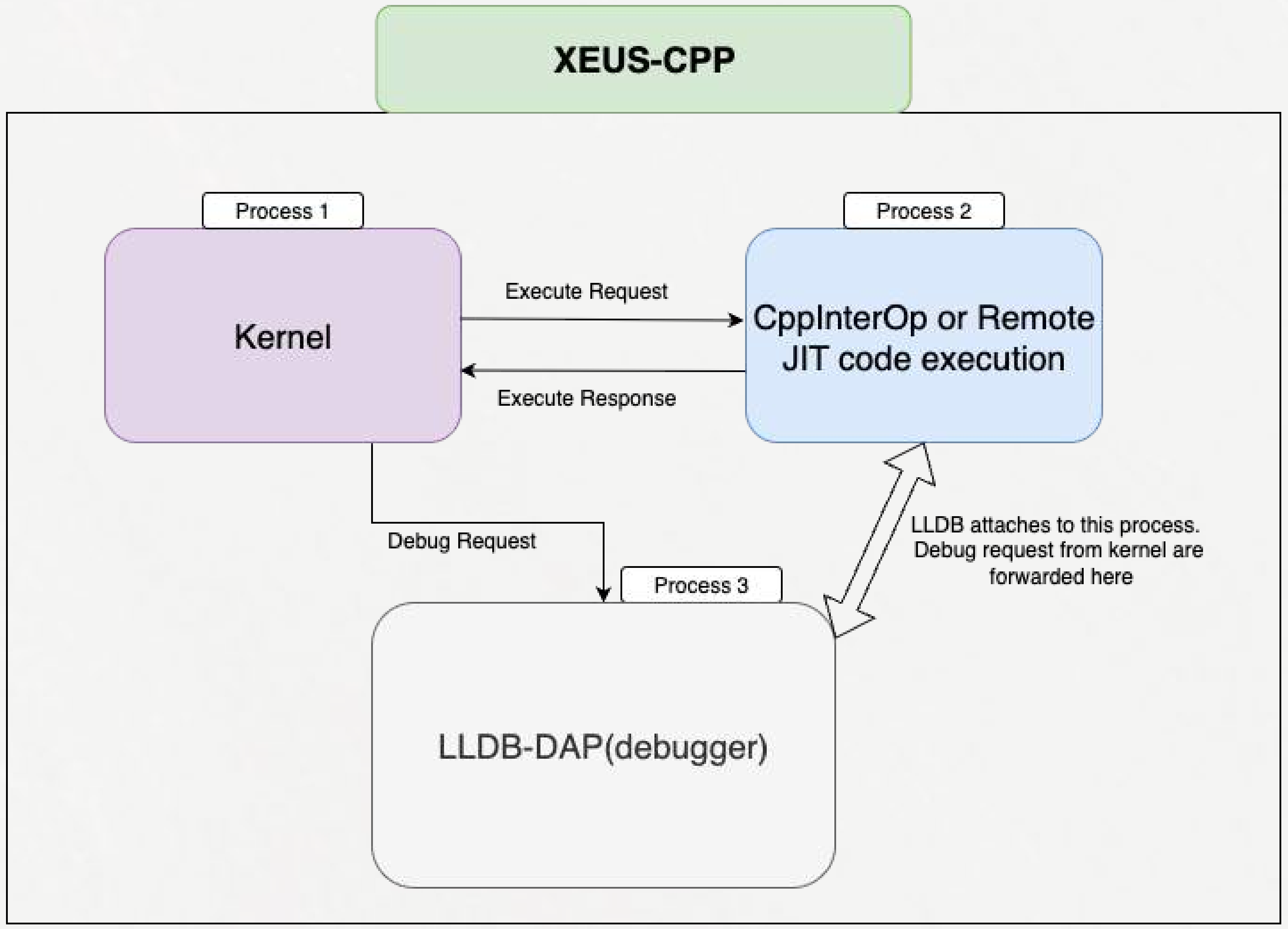
2. It is built on the Xeus library—a C++ implementation of the Jupyter kernel protocol.

3. Powered by the Clang-Repl interpreter from the CppInterOp library, Xeus-Cpp allows you to write, execute in real-time, much like you would with Python.

Why debugger support for xeus-cpp?

- **Missing Critical Feature:** No integrated debugging forces primitive printf-style workarounds
- **C++ Complexity:** JIT-compiled code requires breakpoints and variable inspection for effective bug diagnosis
- **Professional Requirement:** Debugging support is essential to make xeus-cpp production-ready

Workflow


XEUS-CPP
Process 1
Kernel
Execute Request
Execute Response
Process 2
CppInterOp or Remote JIT code execution
Debug Request
Process 3
LLDB-DAP(debugger)
LLDB attaches to this process. Debug request from kernel are forwarded here

Work done in JupyterLab

Pull Request: Fix threadId being passed to the debugger #17667

Identified and fixed a bug in JupyterLab's frontend debugger implementation.
Previously, the DebuggerService::_currentThread method returned a hardcoded value of 1.
This was corrected to dynamically return the first available threadId, ensuring accurate thread handling during debugging sessions.

Issue: Bug: Multiple configurationDone Requests Sent by JupyterLab #17673

Discovered that JupyterLab was sending a configurationDone request after every setBreakpoints call.
According to the Debug Adapter Protocol (DAP) specification, this request should only be sent **once**, after all initial configuration is complete.
This issue was reported and documented for further upstream resolution.

Work done in CppInterOp

Pull Request: Documentation for debugging CppInterOp using LLDB #621

Added comprehensive documentation describing how to debug **CppInterOp** using **LLDB**, making it easier for developers to inspect and troubleshoot CppInterOp internals.

Pull Request: Out-Of-Process Interpreter for CppInterOp #717

Implemented an **Out-of-Process Interpreter** for CppInterOp.
This enhancement utilizes LLVM's llvm-jitlink-executor and the **ORC Runtime** to delegate JIT execution to a separate process.
Users can enable this functionality simply by passing the --use-oop-jit flag as a ClangArg when constructing the interpreter.

Work done in LLVM

Pull Request: [clang-repl] Adds custom lambda in launchExecutor and PID retrieval (Merged but later reverted by #153180)

Introduced:

- A **custom lambda function** in launchExecutor.
- Support for **retrieving the PID** of the launched out-of-process (OOP) JIT executor.

However, due to bot-related infrastructure issues, this PR was later reverted.

Subsequent Pull Requests

- [clang-repl] Sink RemoteJITUtils into Interpreter class (NFC) #155140
- [clang-repl] Add support for running custom code in Remote JIT executor #157358
- [clang-repl] Disable out of process JIT tests on non-unix platforms #159404

These follow-up PRs addressed the functionality of the reverted change by:

- **Refactoring RemoteJITUtils**, creating the JitBuilder inside the Interpreter class.
- **Adding support for custom lambdas** in launchExecutor.
- **Enabling PID retrieval** for the launched OOP JIT executor.
- **Improving test reliability** by disabling OOP JIT tests on non-Unix platforms.

Work done in xeus-cpp

The changes in xeus-cpp are currently awaiting review and merge.

Pull Request: Debugger for xeus-cpp with testing framework #401

This pull request introduces comprehensive debugger support for the xeus-cpp kernel.

Key contributions include:

- A new kernel variant with an out-of-process interpreter and integrated debugger support.
- Integration of LLDB-DAP within the xeus environment.
- A dedicated testing framework to validate and ensure the reliability of debugger functionality.

Demo

Docker Image

The provided Docker image is based on **Ubuntu 22.04 (x86_64)**. You can run it on any x86_64 host machine.
When launched, it automatically starts a **JupyterLab** instance configured with the **xcpp17-debugger** kernel, allowing you to experiment with the debugger directly.

Commands to Run

```
docker pull kr2003/xcpp-debugger
docker run -it --privileged -p 8888:8888 kr2003/xcpp-debugger
```

Once the container starts, open localhost:8888 in your browser to access JupyterLab and try out the debugger.

Video Link

Future Work

- Explore LLDB in WASM and debugger(DAP) support in jupyterlite.
- Extend LLDB-DAP or DAP with more advanced C++ specific debugging techniques. For example, watchpoints, etc.
- Have a robust testing framework for whole xeus-cpp. xeus projects does not have a good testing framework.
- Optimise out-of-process interpreter by using Shared Memory.
- Have out-of-process interpreter extended to different architectures(currently only supported on linux-x86_64 and macos-darwin)
- I would love to contribute more on implementing more features in clang-repl/CppInterOp even after GSoC.

Date
13 November 2025

The End

THANK YOU FOR LISTENING

Thanks Vassil, Vipul and Anutosh!!
Thanks Compiler Research Group!!