The CaaS Project.
Progress & Plans Q2, Q3

Vassil Vassilev

Project Goals

- Support for incremental compilation (clang::libInterpreter, Clang-Repl)
- Language interoperability layer (cppyy, libInterOp)
- Heterogeneous hardware support (offload execution, clad demonstrator)
- Use case development & community outreach (tutorial development, demonstrators)

Project Goals

In [1]:
```
struct S { double val = 1.; };
```

In [2]:
```
from libInterop import std
python_vec = std.vector(S)(1)
```

In [3]:
```
print(python_vec[0].val)
```

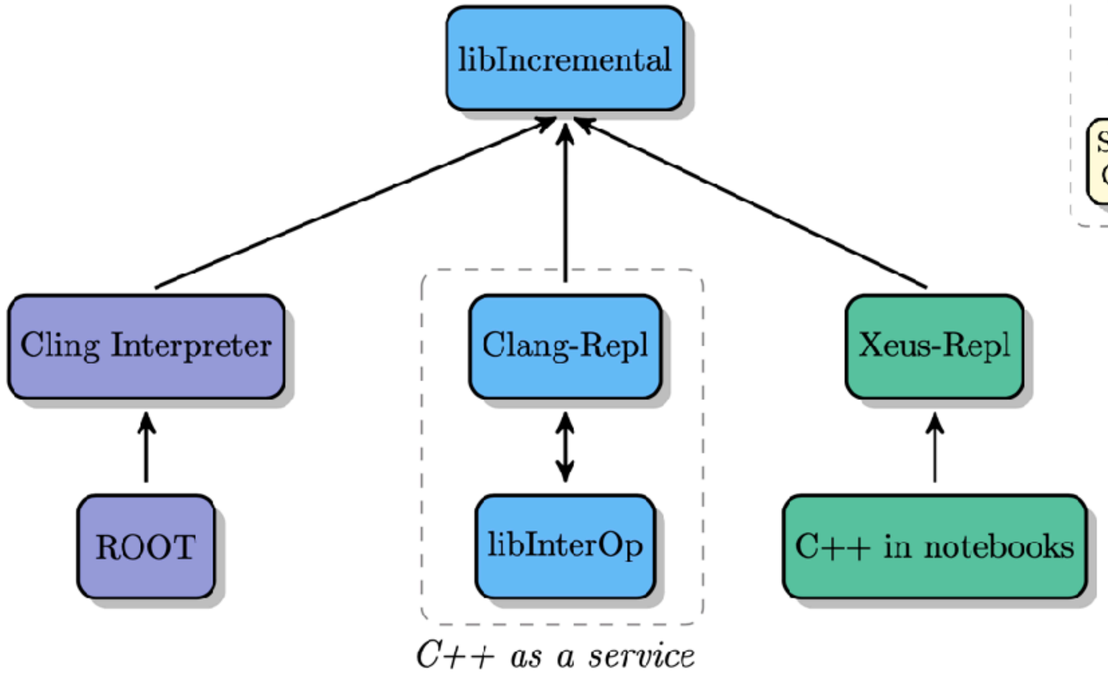
```
1
```

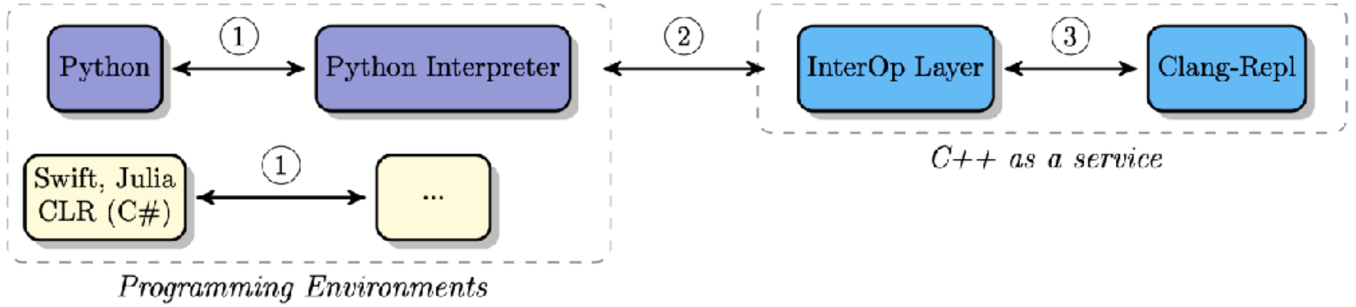
In [4]:
```
class Derived(S)
    def __init__(self):
        self.val = 0
res = Derived()
```

In [5]:
```
__global__ void sum_array(int n, double *x, double *sum) {
  for (int i = 0; i < n; i++) *sum += x[i];
}
// Init N=1M and x[i] = 1.f. Run kernel on 1M elements on the GPU.
sum_array<<<1, 1>>>(N, x, &res.val);
```

Enable bi-directional language communication capable of controlling accelerator hardware

Project Goals





Reroute the cling-based ecosystem more to llvm upstream

Q2 Progress

1. [Q1/VV] Upgrade to LLVM 13 — 90% complete

2. [Q1/VV] Update Cling to use more of LLVM13 — 60% complete (depends on 1.)

3. [Q1/DL] Construct simple patches to upstream dashboard to track — 100% complete

4. [Q1-Q4] Upstream Cling-specific patches — **15/88** complete

5. [Q1-Q4/DL] Keep track of Cling SLoC — Q2
 50 files changed, 695 insertions(+), 1167 deletions(-)

6. [Q2/II] Connect Clang-Repl to the Python Interpreter —70% complete, needs to land in llvm

7. [Q2/PA] Differentiate CUDA kernels — complete for forward mode

8. [Q2/VV] Implement in clang an extension to allow statements on the global scope — D127284

9. [Q2/PC] Advance error recovery and code unloading — D126682

10. [Q4/II/VV] Connect to xeus-cling (scope out missing functionality for xeus-repl) — working Jupyter Xeus-ClangRepl kernel

11. [Q3/II/VV] Develop demonstrators (eg the one we used for the cssi proposal) — simple example based on builtin types.

Carry-over for Q3

1. Rebase cppyy to use cling-only interfaces (making cppyy ROOT-independent) — Q1/BK → Q3
   The task is about transforming the various ROOT Meta layer calls to their underlying clang/cling analogs

2. Define a set of new classes which handle what's needed (eg TClingCallFunc, etc) — Q1/BK → VV/Q3
   The task is about extracting the common cases where we need a lot of boilerplate code and provide abstractions for it. For example, the mechanism to call functions in a uniform way (currently done with TClingCallFunc) needs to modernized into its own ROOT-independent entity in libInterOp

3. Connect libInterOp with clang-repl — Q2/BK → Q3
   The python interpreter provides C API which allows to expose itself and switch to writing python code on the prompt. In ROOT this happens via TPython::Prompt and we want the modern version of this for clang-repl.

4. Improve test cases and demonstrators — Q2/II → Q3
   The task is about updating the existing demonstrators and developing new ones given the advances in Clad.

Carry-over for Q3

1. Add extensible value printing facility — Q2/VV → Q3
   The task is to improve and generalize the implementation of the PTX support in cling and demonstrate it in clang-repl.

2. Rebase cppyy to use clang-repl/libInterpreter interfaces — Q2/BK → Q3

3. Develop demonstrators (eg the one from the Jupyter mockup) — Q2/BK → N/A

4. Design and Develop a CUDA engine working along with C++ mode — Q2/II → N/A
   The task is to improve and generalize the implementation of the PTX support in cling and demonstrate it in clang-repl.

5. Design and implement a backend capable of offloading computations to a GPGPU. Assess technical performance of gradient produced by Clad on GPGPU — Q2/II,VV → N/A

6. Support Tensors and showcase differentiation of Eigen entities — Q1/PA → N/A

Carry-over for Q3

7. Add more clad benchmarks — Q2/DL → Q3
8. Add extensible value printing facility — Q2/VV → Q3
9. Write a paper on incremental C++ — Q2/VV → Q3
10. Write a paper on AD for the aggregate types — Q2/PA → N/A
11. Write an Error Estimation paper — Q2/GS → Q3

Plans for Q3

1. Upstream the type sugaring patch — GSoC Matheus
   The task includes re-engineering the solution we have in ROOT and making it acceptable for Clang.

2. Upstream the lazy template specializations patch — GSoC Tapasweni
   The task includes re-engineering the solution we have in ROOT and making it acceptable for Clang.

3. Develop documentation, examples and tutorials (in llvm documentation as well) — Sara and Rohit
   The task writing technical documentation and blog posts about the developed technologies.

4. Initiate tutorial development within the Clang-Repl community and integrate Clang-Repl into Xeus. Blog post on working notebook demonstrating tutorial — Sara and Rohit?
   The task writing technical documentation and blog posts about the developed technologies.

Plans for Q3

5. Implement an API to offload computations on GPGPUs in CaaS allowing to mix C/C++/CUDA and demonstrate Clad gradient in CUDA — ?

6. Optimize ROOT use of modules for large codebases (eg, CMSSW) — GSoC Jun

   One source of performance loss is the need for symbol lookups across the very large set of CMSSW modules. ROOT needs to be improved to optimize this lookup so that it does not pull all modules defining namespace `edm` on `edm::X` lookups. The task includes implementing a global module index extension which keeps information if an identifier name was a namespace and then integrating it in CMSSW builds.