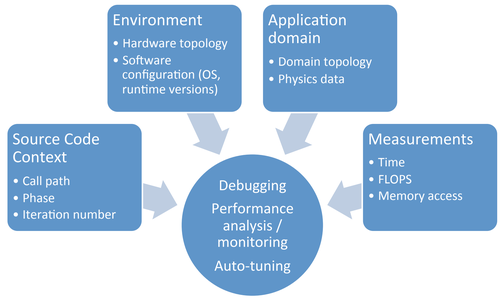
Caliper: Application Introspection for Composite HPC Codes

David Böhme, Todd Gamblin, Peer-Timo Bremer, Olga Pearce, and Martin Schulz
Lawrence Livermore National Laboratory

Tools Depend on Application Introspection



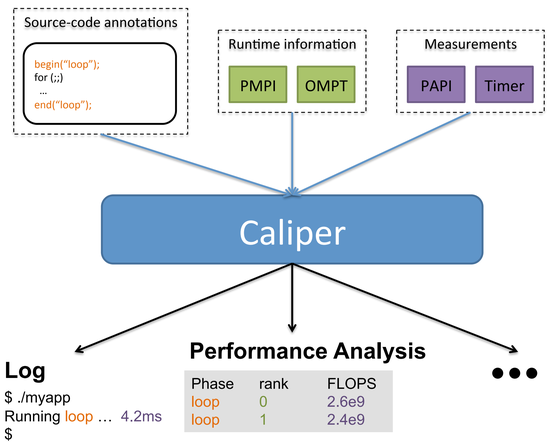
Cross-cutting software engineering tools need to access and correlate context information *(e.g., program phase, iteration number)*, measurements *(e.g., hardware counters, time)* and other data across application modules and threads/tasks: e.g., "how much time was spent in iteration *x*?"

Universal Application Introspection with *Caliper*

Caliper is a framework to collect, combine, and query application context information and (performance) measurement data in HPC applications. Developers provide application context information through an annotation API and measurement data through measurement modules.

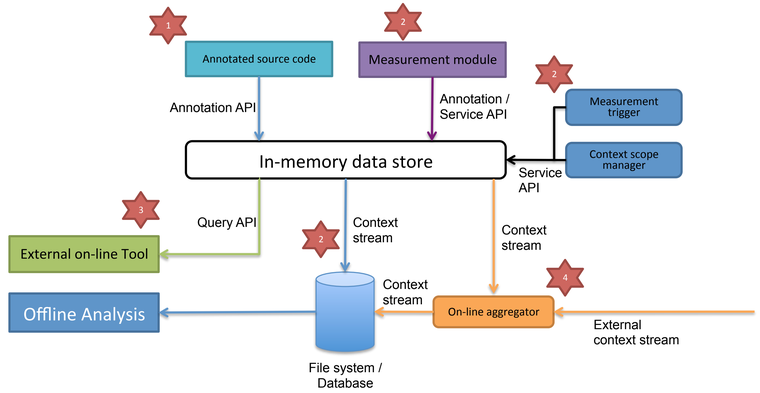
Goals

- **Annotate** components and write measurement modules **independently**,
- Use them in **any combination** for **any purpose**, controlled at runtime



Caliper facilitates re-combination and re-use of source code annotations and measurement modules for different use cases.

Caliper Framework Overview

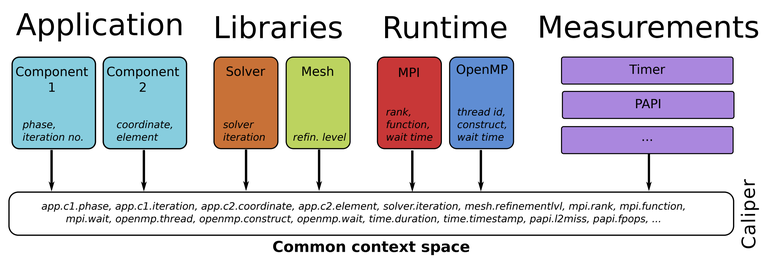


Caliper workflow: (1) Annotated modules independently update context records. (2) Context snapshots write combined context and measurement data into the output context stream. (3) Optional external tools can query the context buffer. (4) Optionally, streams are aggregated on-line to create profiles.

- **Annotation API**
  Provides application context information as *attribute:value* pairs.
- **In-memory data store**
  The process-wide context store combines the attributes set by individual context annotations into a shared, space-efficient *context tree* structure.
- **Measurement modules**
  Provide performance data or runtime context information, such as timers, call path, hardware performance counters (PAPI and PEBS), and MPI and OpenMP runtime context.
- **Tool / query API**
  External tools or measurement services can trigger *context snapshots,* which automatically combine context and measurement data.

Data Composition across the Software Stack

Annotations in different modules across application layers are automatically merged and combined with measurement data.



Annotations and measurement data from different modules are combined into a common context space.

Context Annotation API

C and C++ APIs are available to provide context information.

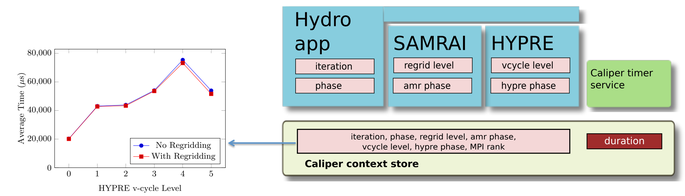
```
#include <Annotation.h>

int main(int argc, char* argv[])
{
    cali::Annotation phase_ann("phase");

    phase_ann.begin("init");
    // Perform initialization
    initialize();
    phase_ann.end(); // ends "init"

    phase_ann.begin("loop");

#pragma omp parallel for
   for (int i; i < MAX; ++i) {
        cali::Annotation("iteration").set(i);
        do_work(i);
    }

    phase_ann.end(); // ends "loop"
}
```

C++ API example. Application developers can specify hierarchical (*phase* attribute), scalar (*iteration* attribute) or user-defined context information.

Case Study: LLNL hydrodynamics code

We used Caliper to instrument an LLNL radiation hydrodynamics code that uses the AMR library SAMRAI and the linear solver library HYPRE.

- Separate independent annotations added in the program and libraries combined with measurement data enable analysis of the interactions between the components.



Composition of measurement data and application annotations from the main application, HYPRE, and SAMRAI allows analysis of the interactions between components - e.g., study the influence of AMR regrid cycles on time spent in the HYPRE solver.

amr_phase=regrid/loop				duration=76019
amr_phase=regrid/loop	regrid_level=1			duration=40
amr_phase=regrid/loop				duration=6
amr_phase=regrid				duration=361714
amr_phase=regrid			phase=main/loop	duration=143816
amr_phase=regrid		hypre_phase=vcycle	phase=main/loop	duration=20
amr_phase=regrid		hypre_phase=vcycle/loop	phase=main/loop	duration=435
amr_phase=regrid	vcycle_level=1	hypre_phase=vcycle/loop	phase=main/loop	duration=358
amr_phase=regrid	vcycle_level=2	hypre_phase=vcycle/loop	phase=main/loop	duration=146
amr_phase=regrid	vcycle_level=3	hypre_phase=vcycle/loop	phase=main/loop	duration=64

Excerpt of Caliper output of instrumentation in the main application (phase), SAMRAI (amr_phase and regrid_level), and HYPRE (hypre_phase and vcycle_level), with time information provided by timer service.

Caliper is available on github:
https://github.com/scalability-llnl/Caliper

Contact: boehme3@llnl.gov

This work performed under the auspices of the U.S. Department of Energy by Lawrence Livermore National Laboratory under Contract DE-AC52-07NA27344

LLNL-POST-676083