Advancing Performance Tools Through Collaboration With STEP

Oak Ridge National Laboratory, University of Tennessee
Terry Jones, Michael R. Jantz

STEP
SOFTWARE TOOLS
ECOSYSTEM PROJECT

Terry Jones – STEP Director
trj@ornl.gov
Oak Ridge National Laboratory

Michael R. Jantz – Deputy Director
mrjantz@utk.edu
University of Tennessee

Abstract

The Software Tools Ecosystem Project (STEP) is a DOE initiative to provide the ***ongoing*** support and enhancement needed for critical High Performance Computing (HPC) software tools to remain ***effective, efficient, and relevant*** in the rapidly evolving field of HPC. We define Tools to mean **the collection of software packages that can be applied to monitor, analyze, and diagnose performance and behavior of computational science applications and systems**.

Motivation

Software tools play an essential role in optimizing scientific application performance and maximizing resource efficiency. In addition to enabling supercomputer performance (a decisive determinant of scientific discovery), these tools provide feedback to users, operations staff, and software developers that amplifies the long-term impact of investments in scientific computing. **A unified tools project not only supports individual software teams but also fosters a healthy ecosystem for collaboration.**

Approach

- Identify emerging platform and application needs and key gaps tool functionality.
- Apply novel software development strategies to extend and enhance software tool capabilities.
- Pursue opportunities for co-design with hardware vendors, application developers, facilities and tool developers.

Challenges

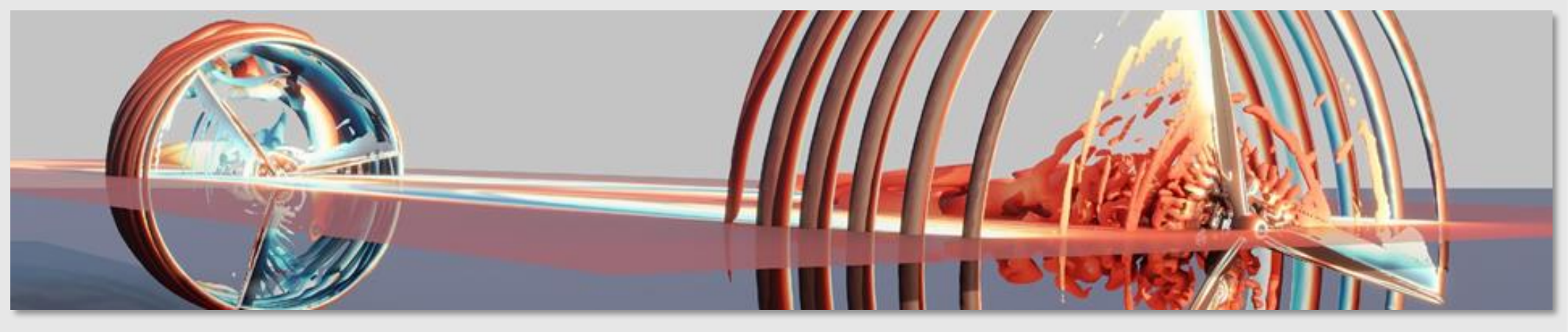
- Rapidly changing technologies
- Steady increase in system complexity
- Software diversity in OS versions, programming languages, environments, schedulers, and libraries
- CPU and GPU diversity

Testimonials

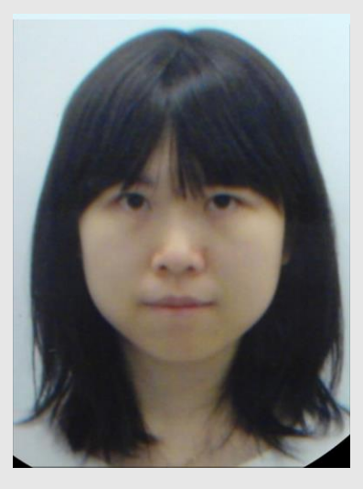
HPCToolkit

I just wanted to mention we've been using HPCToolkit a lot for our ExaWind application on Frontier, which is a hugely complicated code, and your profiler is one of the only ones we've found that really lets us easily instrument and then browse what our application is doing at runtime including GPUs. As an example, during a recent hackathon we had, ***we improved our large scale performance by 24x*** by understanding our code better with HPCToolkit and running it on 1000s of nodes while profiling. We also recently improved upon this by 10% for our total runtime.

– Jon Rood NREL (5/31/2024)

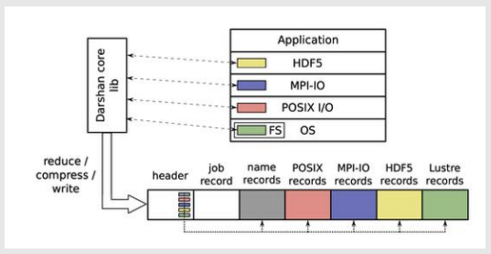


Darshan



The Darshan I/O characterization tool has provided key insights into High Energy Physics (HEP) experiments such as ATLAS. For example, collaboration between our teams in the HEP-CCE (High-Energy Physics Center for Computational Excellence) project has led to **new insights that enable more efficient management of the exabyte-scale data produced by the ATLAS experiment**, with Darshan data helping drive critical data management design decisions for the ATLAS Athena software framework. Furthermore, practical experience with using Darshan in this framework has led to key enhancements to Darshan that should help broaden its impact to other software frameworks employing complex multi-process workflows.

– Rui Wang, ANL (Mar-13-2025)

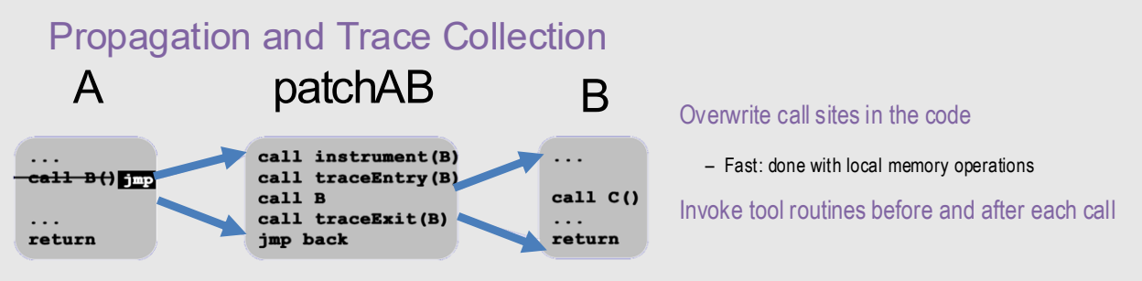


High-level overview of Darshan's modularized runtime architecture and self-describing log file format

Dynist

The HPCToolkit performance tools, used by scientists for pinpointing and diagnosing performance and scalability problems in applications executing on exascale supercomputers, rely on the Dyninst infrastructure developed by the University of Wisconsin - Madison. HPCToolkit uses Dyninst to analyze CPU and GPU binaries to attribute metrics back to application source code. Without Dyninst, HPCToolkit would be unable to attribute performance to inlined code or loop nests in CPU and GPU code. **Such attribution is critical for understanding** the performance of modern template-based programming models, such as Sandia's Kokkos and LLNL's Raja.

– John Mellor-Crummey, Rice (3/16/2025)



```
...
~~call B()~~
...
return
```

```
call instrument(B)
call traceEntry(B)
call B
call traceExit(B)
jmp back
```

```
...
call C()
...
return
```

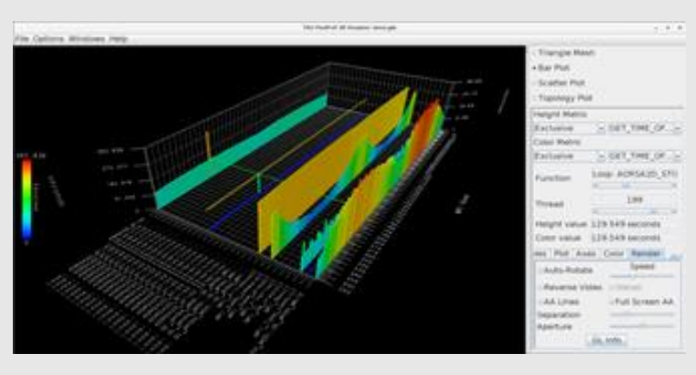
Overwrite call sites in the code
- Fast: done with local memory operations

Invoke tool routines before and after each call

TAU

Sarah Elliott, a Computational Chemist at Argonne National Laboratory, ***improved the total runtime of her application MESS by a factor of 3.8x in one day*** using the TAU performance toolset.

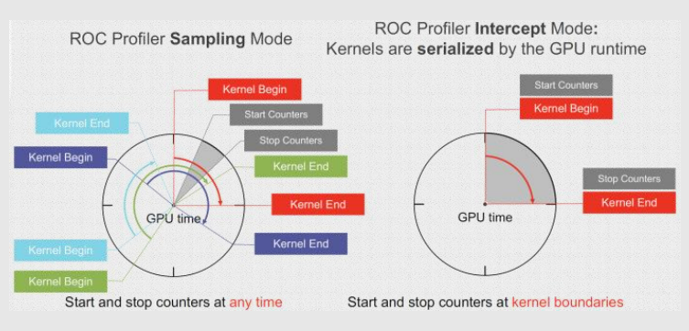
– work performed by *Sarah Elliott, ANL*



PAPI

The Performance Application Programming Interface (PAPI) provides an important "standardized" API for accessing performance counters across a wide variety of hardware components from different vendors. By using this kind of universal layer, performance tools and other software desiring access to these counters can be written in a common fashion, ***making the addition of support for new hardware components far easier*** than directly using vendor provided APIs. PAPI enables this while adding minimal overhead for accessing these counters.

– Larry Kaplan HPE (3/13/2025)

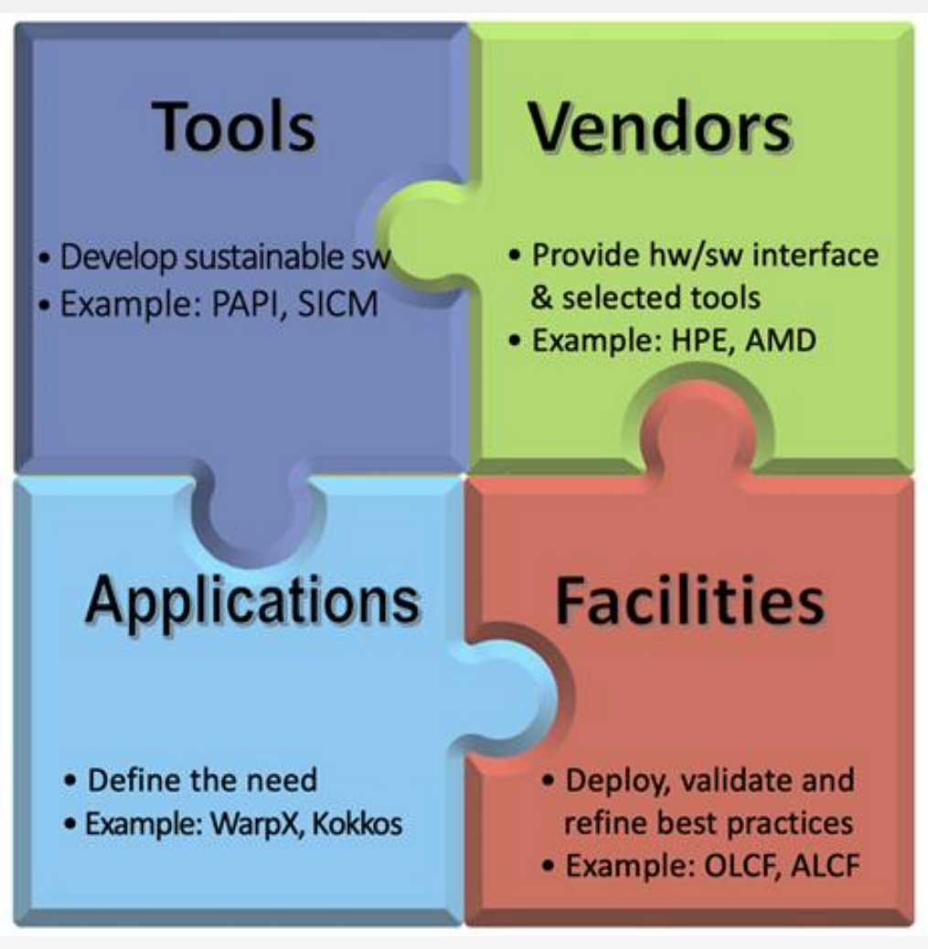

ROC Profiler **Sampling** Mode
ROC Profiler **Intercept** Mode:
Kernels are **serialized** by the GPU runtime
Kernel Begin
Start Counters
Stop Counters
Kernel End
Kernel End
Kernel Begin
Kernel End
Kernel End
Kernel Begin
Kernel Begin
GPU time
Start Counters
Kernel Begin
Stop Counters
Kernel End
GPU time
Start and stop counters at any time
Start and stop counters at kernel boundaries

Potential Impact

- STEP can be a vehicle to improve communication among tool developers, application teams, vendors and facilities.
- STEP can be a clearing-house for efficiently managing critical HPC technology.
- STEP can be a resource for workforce development.

Collaboration Opportunities

- Working together to develop a proactive stance on the rapidly evolving hardware landscape: new (completely different) tool APIs from AMD and Intel that include support for GPU PC sampling
- Working with vendors to help design their tool APIs, such as AMD's emerging rocprofiler-sdk and Intel's emerging PTI-GPU.
- Working with facilities and application teams to understand their emerging tool needs.



STEP SOFTWARE PACKAGE LEADS
Heike Jagode, University of Tennessee
John Mellor-Crummey, Rice University
Barton Miller, University of Wisconsin
Sameer Shende, University Oregon
Wei-Keng Liao, Northwestern University

STEP OVERSIGHT COUNCIL
Terry Jones, Oak Ridge National Lab
Michael Jantz, University of Tennessee
James Brandt, Sandia National Laboratories
James Custer, Hewlett Packard Enterprise
Ann Gentile, Sandia National Laboratories
Richard Gerber, Lawrence Berkeley Natl Lab
Kevin Harms, Argonne National Lab
Heike Jagode, University of Tennessee
Matthew Legendre, Lawrence Livermore Natl Lab
Wei-Keng Liao, Northwestern University
John Linford, NVIDIA Corporation
Keith Lowery, Advanced Micro Devices
Verónica Melesse Vergara, Oak Ridge Natl Lab
John Mellor-Crummey, Rice University
Barton Miller, University of Wisconsin
José Moreira, International Business Machines
Erdal Mutlu, Pacific Northwest Natl Lab
Suzanne Parete-Koon, Oak Ridge Natl Lab
Sameer Shende, University Oregon
Galen Shipman, Los Alamos National Lab
Devesh Tiwari, Northeastern University
Theresa Windus, Ames National Lab

STEP & CASS
To realize our community vision, we are banding together with others to provide crosscutting capabilities related to scientific software stewardship. STEP is one of the founding members of the *Consortium for the Advancement of Scientific Software* (CASS), an organization committed to this vision. To learn more about CASS, please refer to their website at: https://cass.community

DOI: 10.1145/3750720.3757305
Speaker: Terry Jones, trj@ornl.gov
Date: January 26-29, 2026
Venue: SCA/HPCAsia'26, Osaka Japan

This work was supported by the U.S. Department of Energy, Office of Science, Advanced Computing Research, through the Next-Generation Scientific Software Technologies (NGSST) under contract DE-AC05-00OR22725

this poster