Parallel Program Analysis Framework for the ACTS Toolkit

Allen D. Malony
Janice E. Cuny

Computational Science Institute
Department of Computer and Information Science,
University of Oregon

DOE 2000 PI Meeting
http://acts.nersc.gov/pi99

Parallel Program Analysis Framework for the ACTS Toolkit

- ❑ Parallel performance analysis
- ❑ Program code analysis
- ❑ Analysis tool interoperation framework
- ❑ Runtime tool interaction framework

Parallel Performance Analysis

- Tuning and Analysis Utilities (TAU)
- Program Database Toolkit (PDT)
- TAU Monitoring Framework
- Interoperation and Linking Architecture (INTERLACE)

Our Approach

- designing the TAU performance monitor for a general parallel program execution model based on HPC++;
- building a sophisticated program analysis system based on state-of-the-art parsers for C, C++, and Fortran; and
- defining interoperation and interactions methods for connecting tools and programs to one another.

Accomplishments - 1998

Our goals were ...

- To develop a robust portable performance profiling capability for C, C++, and Fortran codes; and
- To integrate performance profiling throughout the ACTS toolkit.

TAU: Tuning and Analysis Utilities

Tau is...

- an extensible tool framework supporting tool interactions with the program, the user, and each other

- a graphical, program development environment with several distinct and unique, but completely integrated, tools

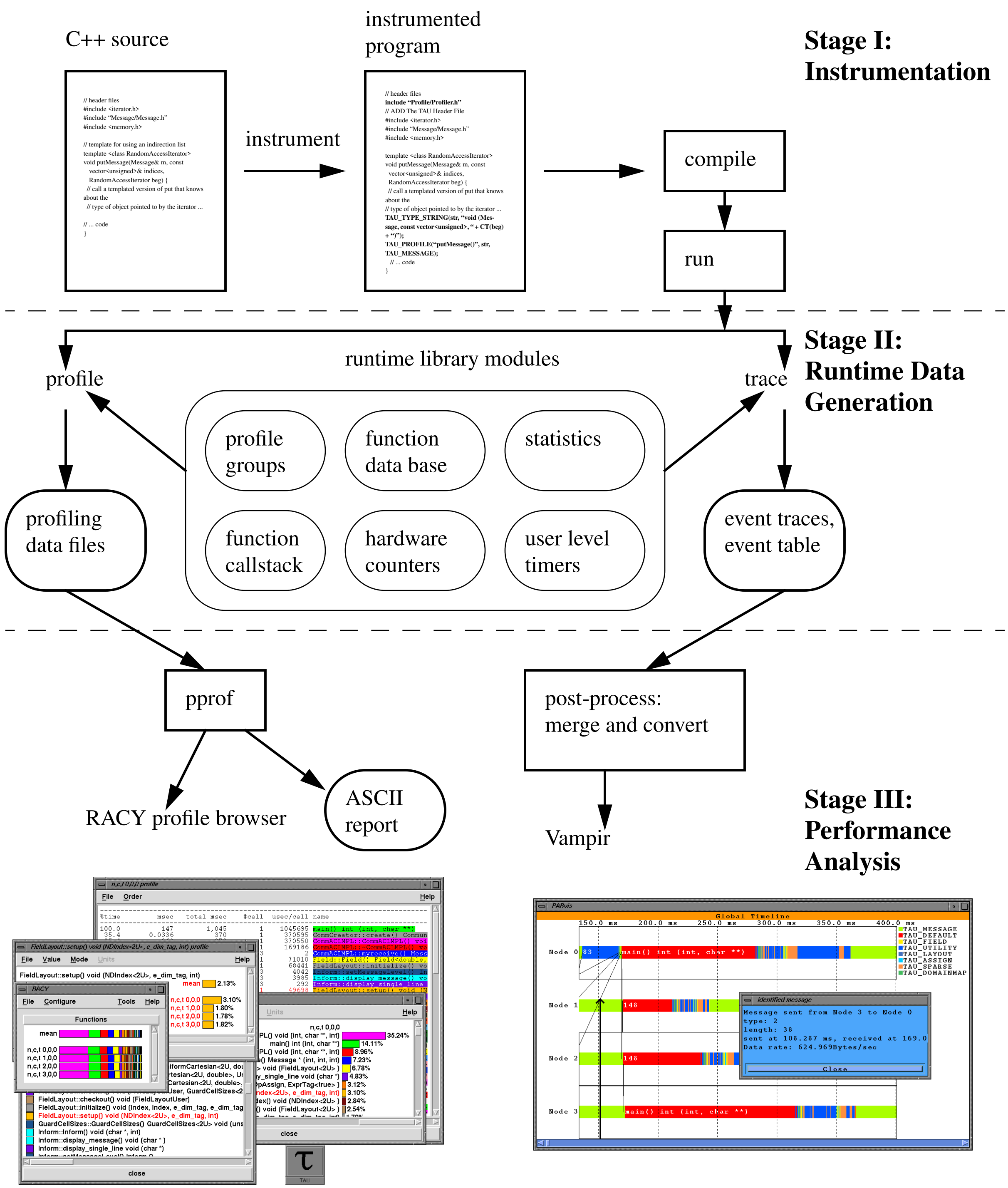
- a performance analysis environment facilitating *static* and *dynamic* interactions with programs

Tuning and Analysis Utilities

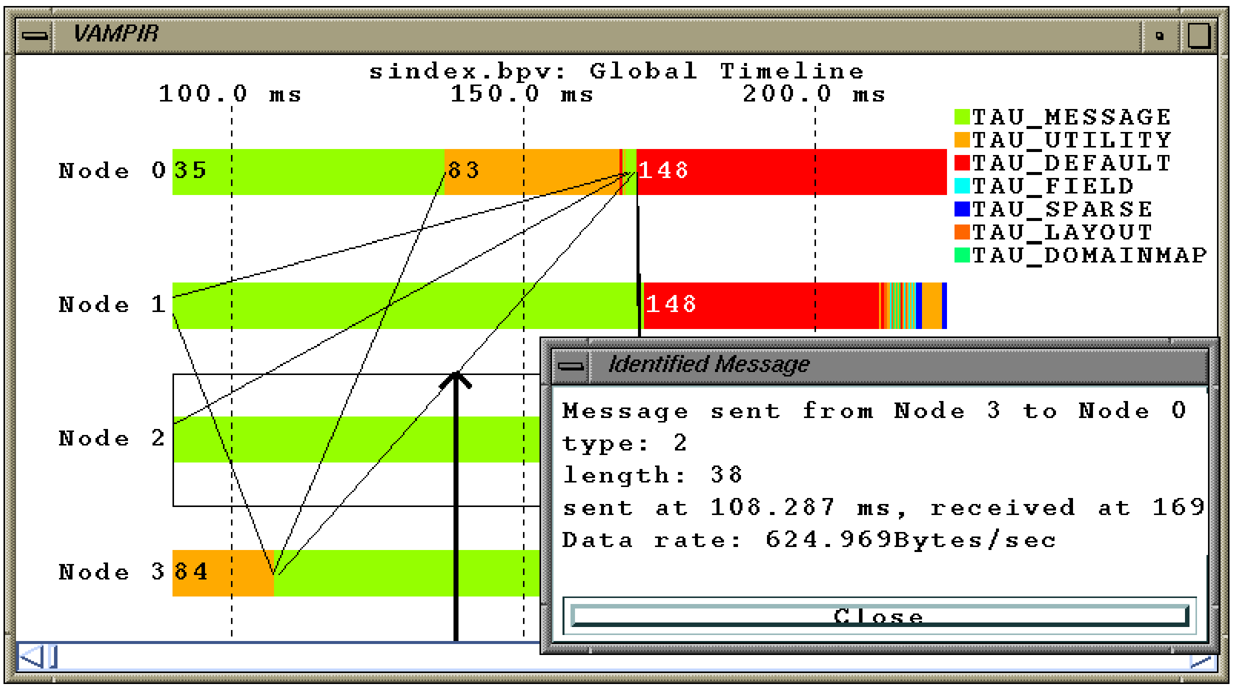
TAU Portable Profiling

- portable profiling and tracing facility for ISO/ANSI C++
  - multiple compilers, operating systems and platforms
  - templates
  - user-defined and statement level timers
  - pthreads, Tulip, and SMARTS Threads supported
- modular and configurable profiling infrastructure
- selective profiling - limit overhead to groups of profiled functions selected at runtime
- lightweight profiling facility, minimal intrusion
- timing-based instrumentation (not sampling) and support for hardware performance counters
- profiling in the presence of optimizations
- high level view of performance data

TAU Portable Profiling and Tracing

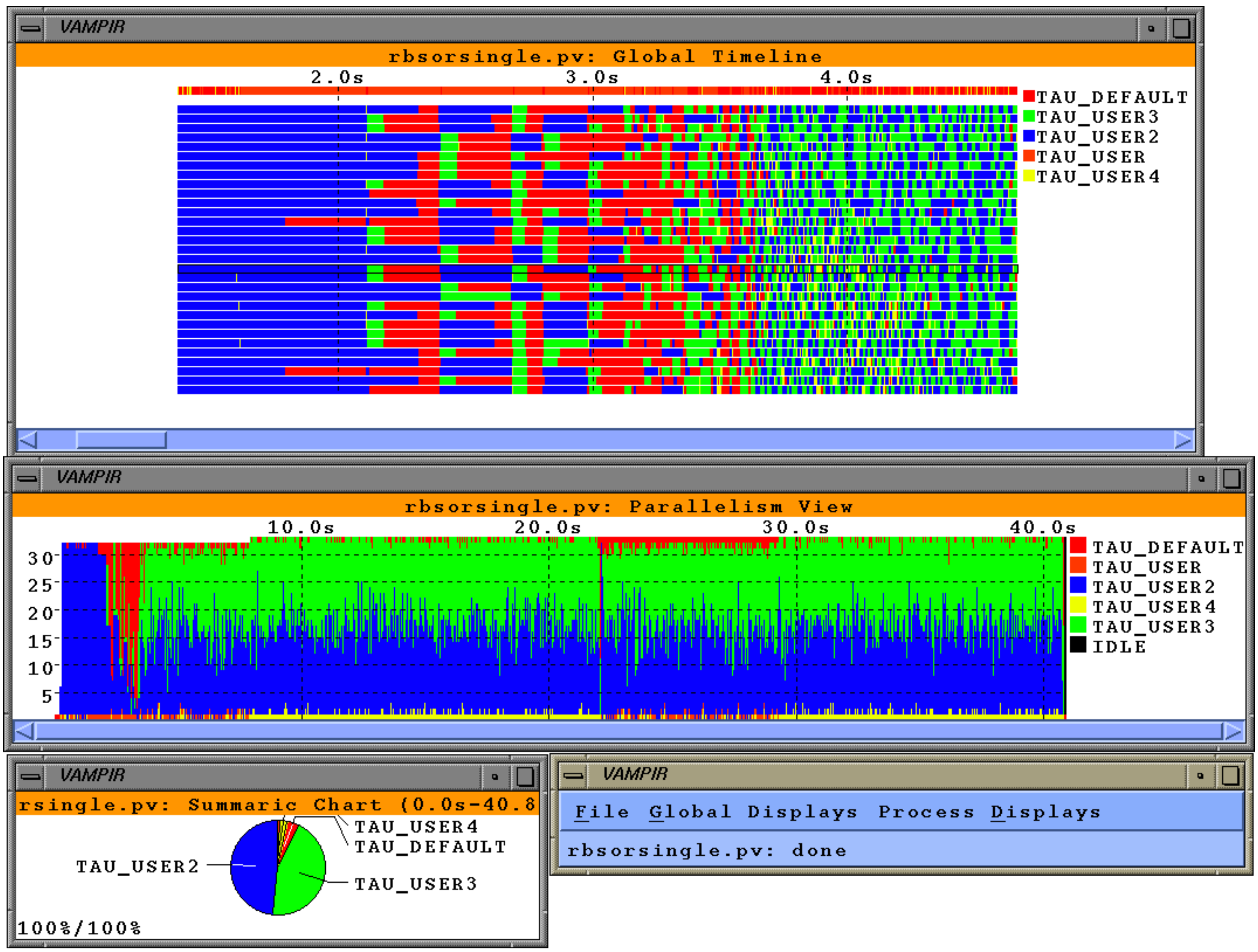


Tracing POOMA Applications



Vampir is from Pallas GmbH & KFA Germany

Tracing SMARTS (Shared Memory Asynchronous Runtime System)



PDT: Program Database Toolkit

The Program Database Toolkit was developed to:

- analyze application source code components
- make this compile-time knowledge available in a uniform way to other tools

Application of PDT

- documentation of program components can be automatically kept accurate and up-to-date
- insertion of instrumentation for performance profiling and tracing can be automated to simplify tuning and result in fewer mistakes
- automated access to interface details when calling library routines can ease programming and eliminate errors
- integration of multi-language programs can be automated, ensuring proper handling for the forms of arguments and return values

PDT Processing Steps

Edison Design Group (EDG) Front End:

- ❑ parses a C++ source file
- ❑ creates an intermediate-language (IL) tree

IL Analyzer:

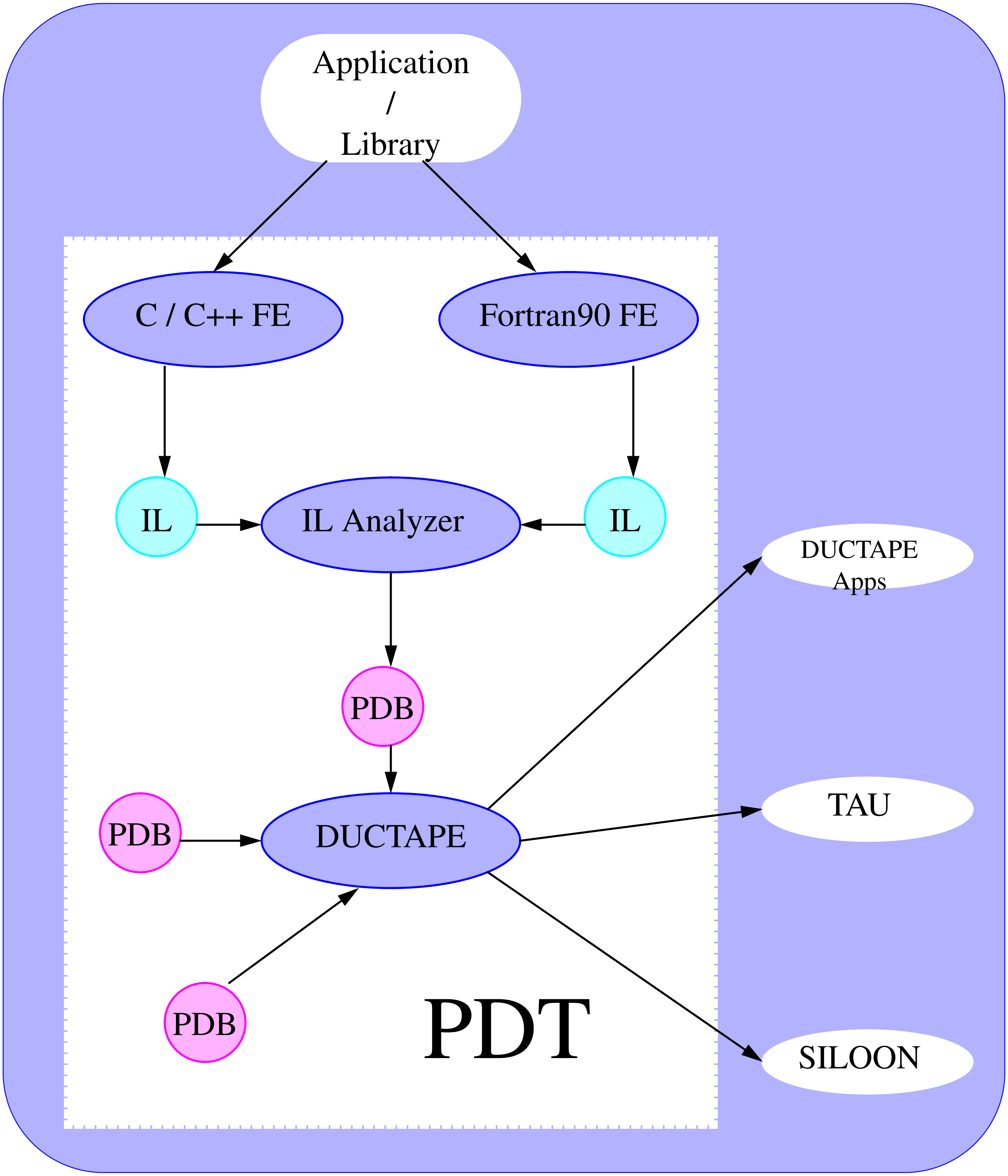
- ❑ processes the intermediate language tree
- ❑ creates another file in "program database" (PDB)

- ❑ format, that can be easily and efficiently read by a programming or a scripting language

DUCTAPE (C++ program Database Utilities and Conversion Tools APplication Environment):

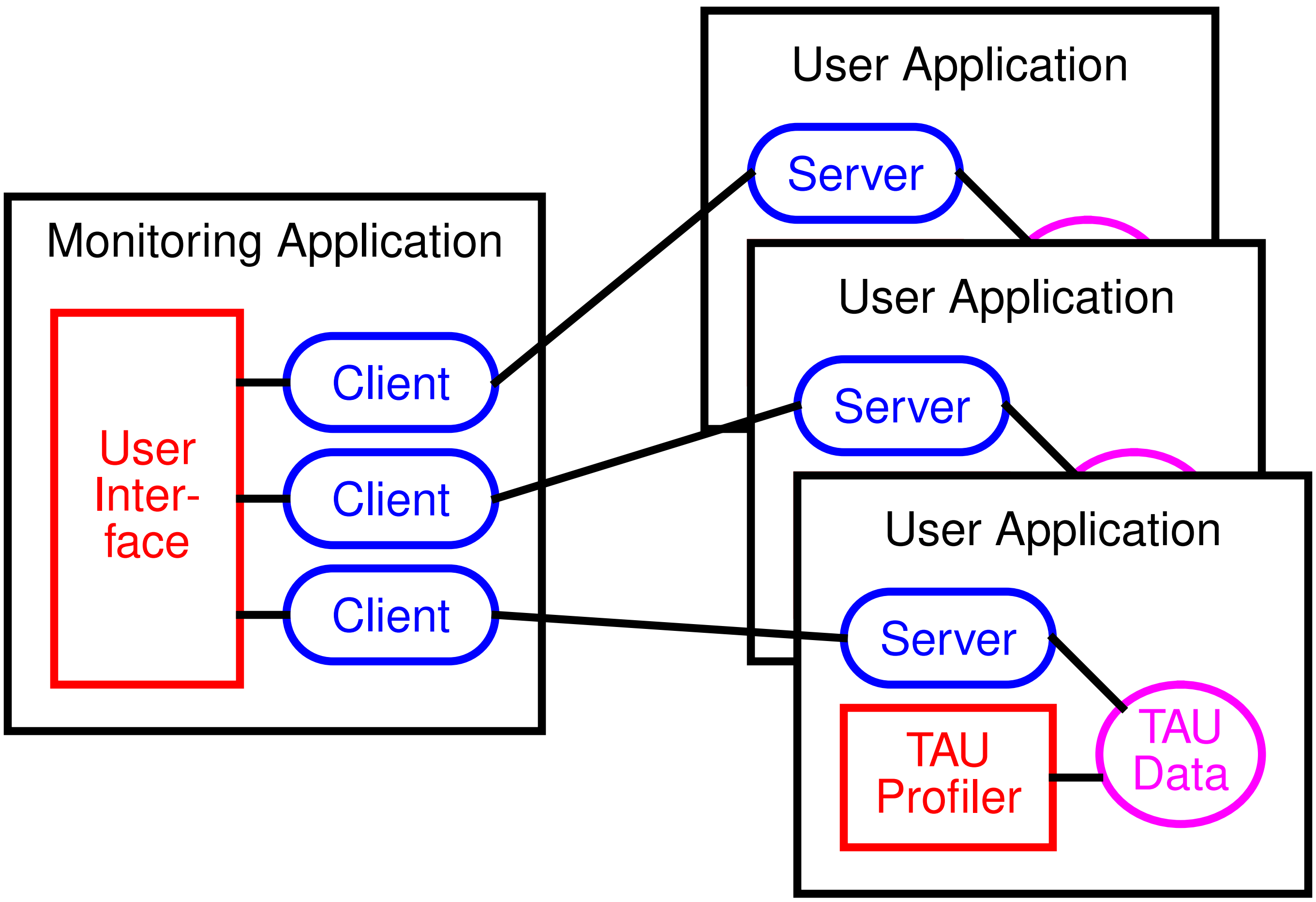
- ❑ processes and merges PDB files
- ❑ is a C++ library that provides access to the "program database" for applications

Program Database Toolkit

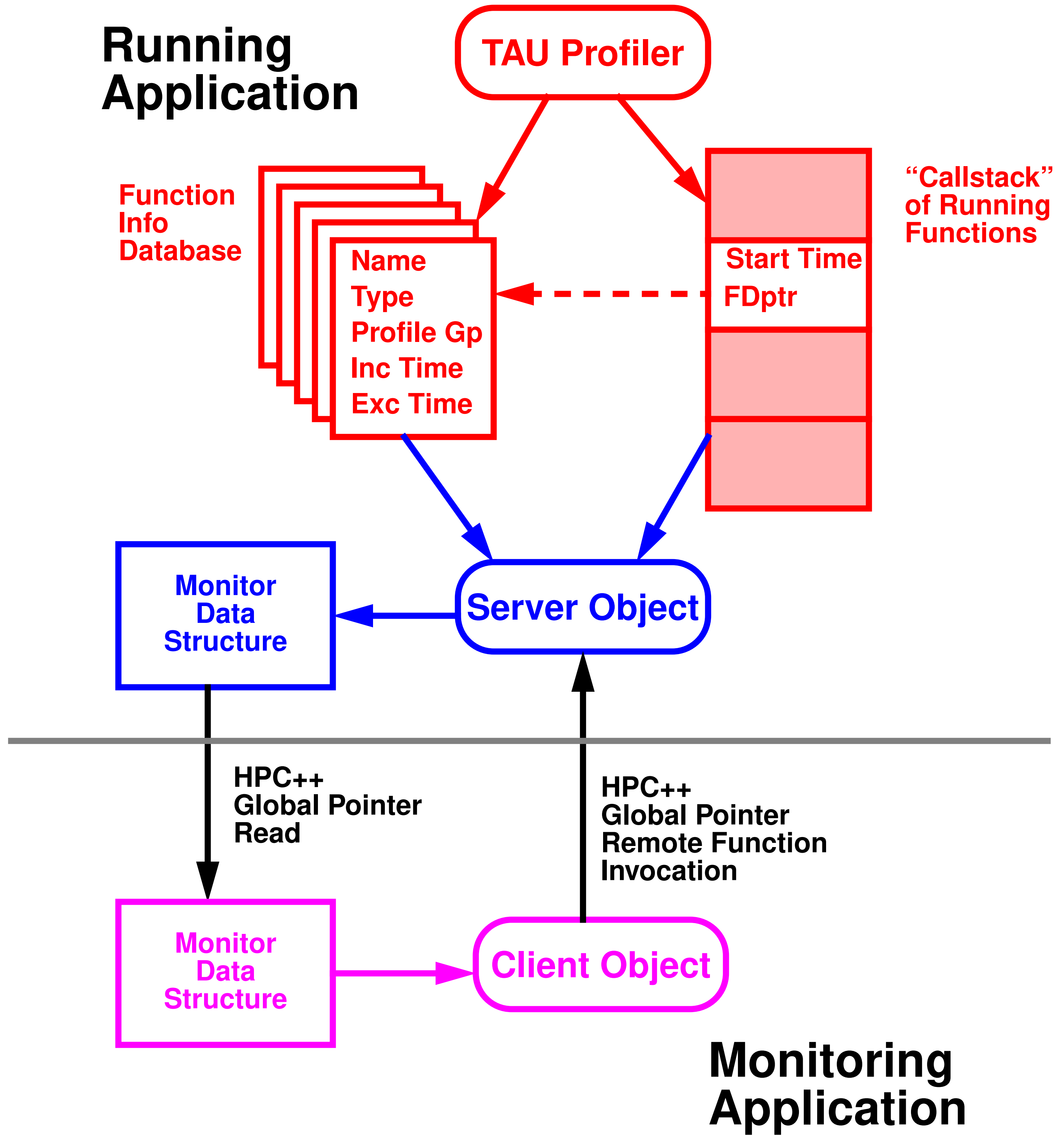


TAU Monitoring Framework

- An object-oriented framework for accessing TAU profiling data from a running application.
- Client-server architecture based on HPC++.
- All aspects of communication encapsulated in objects and hidden from user.



TAU Monitoring Framework - Details



INTERLACE

- An INTERoperation and Linking Architecture for Computational Engines
- Provides a framework for integrating computational engines into an abstract programmable environment.
- Built using HPC++ runtime system
- Prototype computational server implemented using MatLab as a target engine
- Runs on SGI IRIX
- Uses HPC++ for data movement and remote method invocation
- Provides abstract layer for connecting servers and clients to hide platform and engine specific details

INTERLACE Future Work

- Create and implement a class library for basic computational server and wrapper tasks
- Demonstrate engine interoperability by implementing IDL and Mathematica based INTERLACE servers
- Implement extended registry service for dynamic server allocation by clients
- Port current implementation to Linux-based cluster environment
- Examine methods for controlling programming servers, such as GUI based or scripting language based methods

Tool Availability

- http://www.acl.lanl.gov/tau
- http://www.acl.lanl.gov/software
  (ACL Fall 1998 CD-ROM)