The Score-P Community Project
An Interoperable Infrastructure for HPC Performance Analysis Tools

Andreas Knüpfer, Felix Wolf, Sameer Shende

Virtual Institute High Productivity Supercomputing (VI-HPS)

Productivity for Open, INtegrated Tools (POINT)

Overview

- **Introduction and Motivation** *by Andreas Knüpfer*
- Performance Tools Overview *by Felix Wolf*
- Score-P and Tools Demo *by Sameer Shende*
- Conclusions and Discussion

- HPC performance tools are essential
- Several performance tools co-exist
  - Complementary features and overlapping functionality
  - Separate measurement systems and data formats
- Downsides
  - Redundant efforts for development, maintenance, support, training, installation, updates, …
  - Unfavorable user experience, limited interoperability

Plan for Common Infrastructure

- Community effort for a common tools infrastructure
  - Starting with Scalasca, Periscope, TAU, and Vampir
  - Open for other tools and groups

- SILC project (Scalable Infrastructure for Automatic Performance Analysis of Parallel Codes) funded by BMBF, Germany
- PRIMA project (Performance Refactoring of Instrumentation, Measurement, and Analysis Technolgies for Petascale Computing) funded by DOE, US

- First public pre-release at SC‘11

Partners

- Forschungszentrum Jülich, Germany
- German Research School for Simulation Sciences, Aachen, Germany
- Gesellschaft für numerische Simulation mbH Braunschweig, Germany
- RWTH Aachen, Germany
- Technische Universität Dresden, Germany
- Technische Universität München, Germany
- University of Oregon, Eugene, USA

Goals

- Measurement functionality for HPC performance tools
- Support all fundamental concepts
  - Instrumentation (various methods), later sampling
  - Event trace recording
  - Basic and advanced profile generation
  - Online access to profiling data and execution control
- MPI , OpenMP, and hybrid parallelism (and serial)
- Enhanced functionality
  - OpenMP 3.0
  - CUDA, OpenCL, …

- Analysis tools stay separate software packages on top

Requirements

- Functional requirements
  - Call-path profiles and event traces
  - Direct instrumentation, later also sampling
  - All common performance metrics: run-time, visits, communication data, hardware counters, etc.
  - Access and reconfiguration at run-time

- Non-functional requirements
  - Portable to all major HPC platforms, robust, stable
  - Peta-scale level and beyond
  - Low measurement overhead
  - Open source with New BSD license

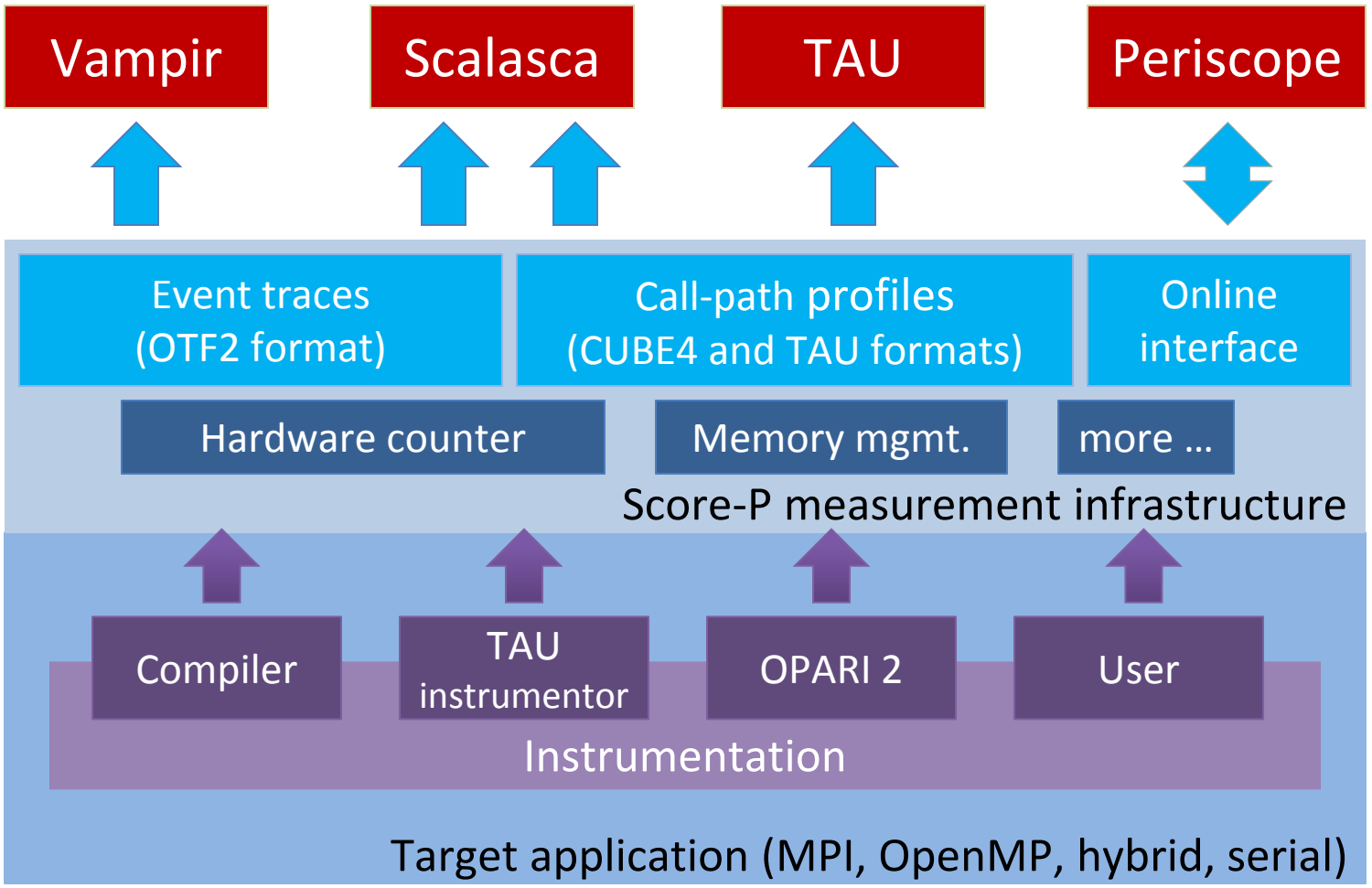
Advantages

- Advantages from developer perspective:
  - Save manpower by sharing development resources
  - Invest in new analysis functionality and scalability
  - Save efforts for maintenance, testing, porting, support, training

- Advantages from user perspective:
  - Single learning curve
  - Single installation, fewer version updates
  - Interoperability and data exchange

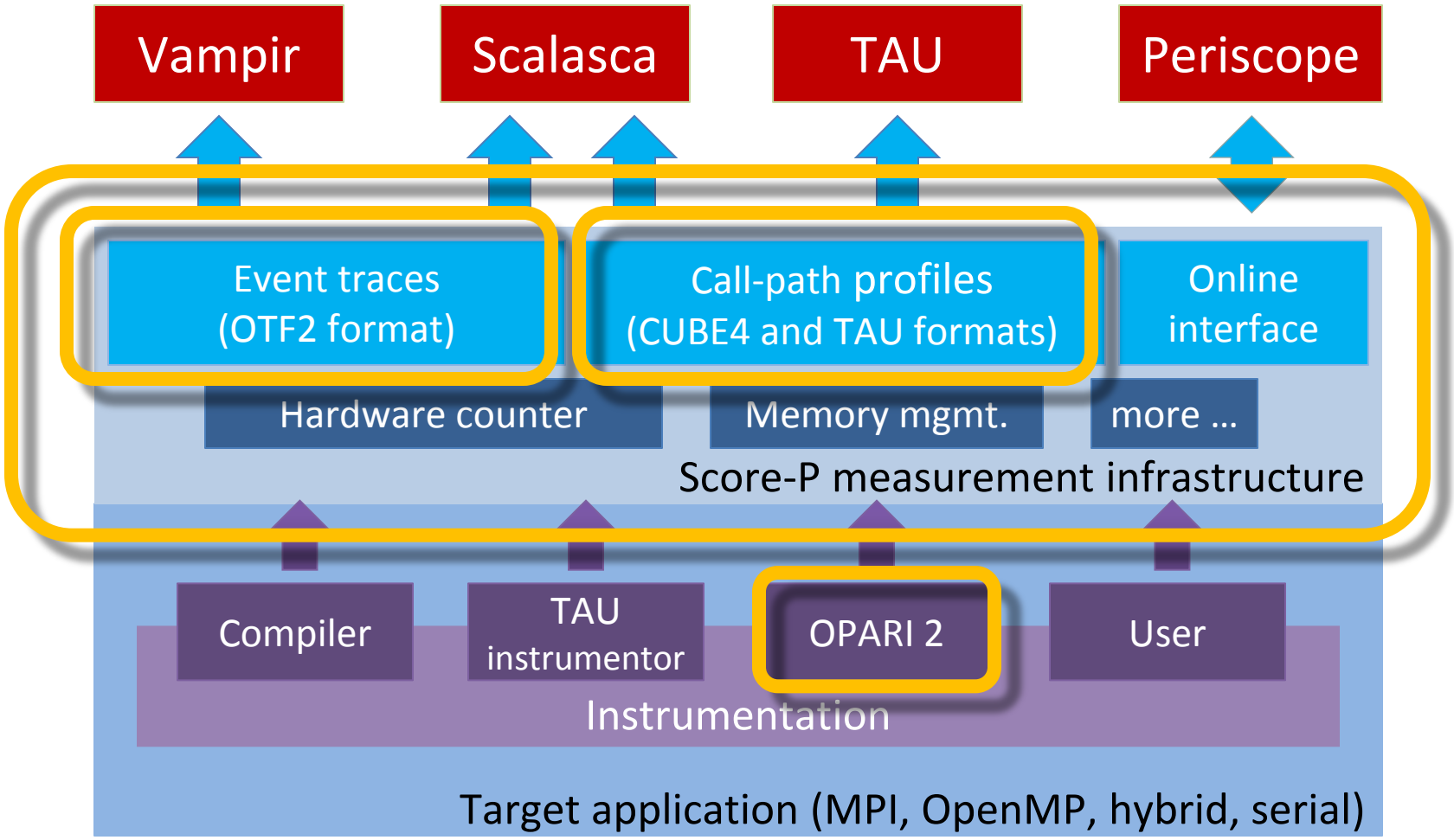
Architecture and Components

- The Score-P software architecture:



Architecture and Software Components

- Separate software components:



Overview

- Introduction and Motivation *by Andreas Knüpfer*
- **Performance Tools Overview** *by Felix Wolf*
- Score-P and Tools Demo *by Sameer Shende*
- Conclusions and Discussion

Initially, the following performance tools will support Score-P

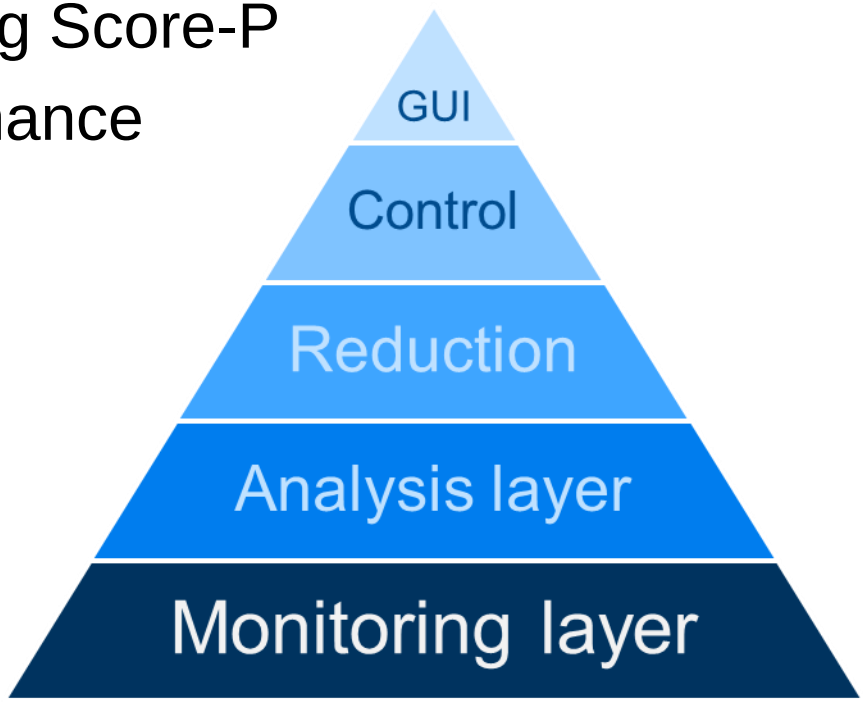
- Periscope – online bottleneck search
- Scalasca – call path profiling and wait-state analysis
- Vampir – trace visualization
- TAU – profile analysis including data mining

Periscope

Performance analysis tool automatically searching for inefficiencies in large scale applications

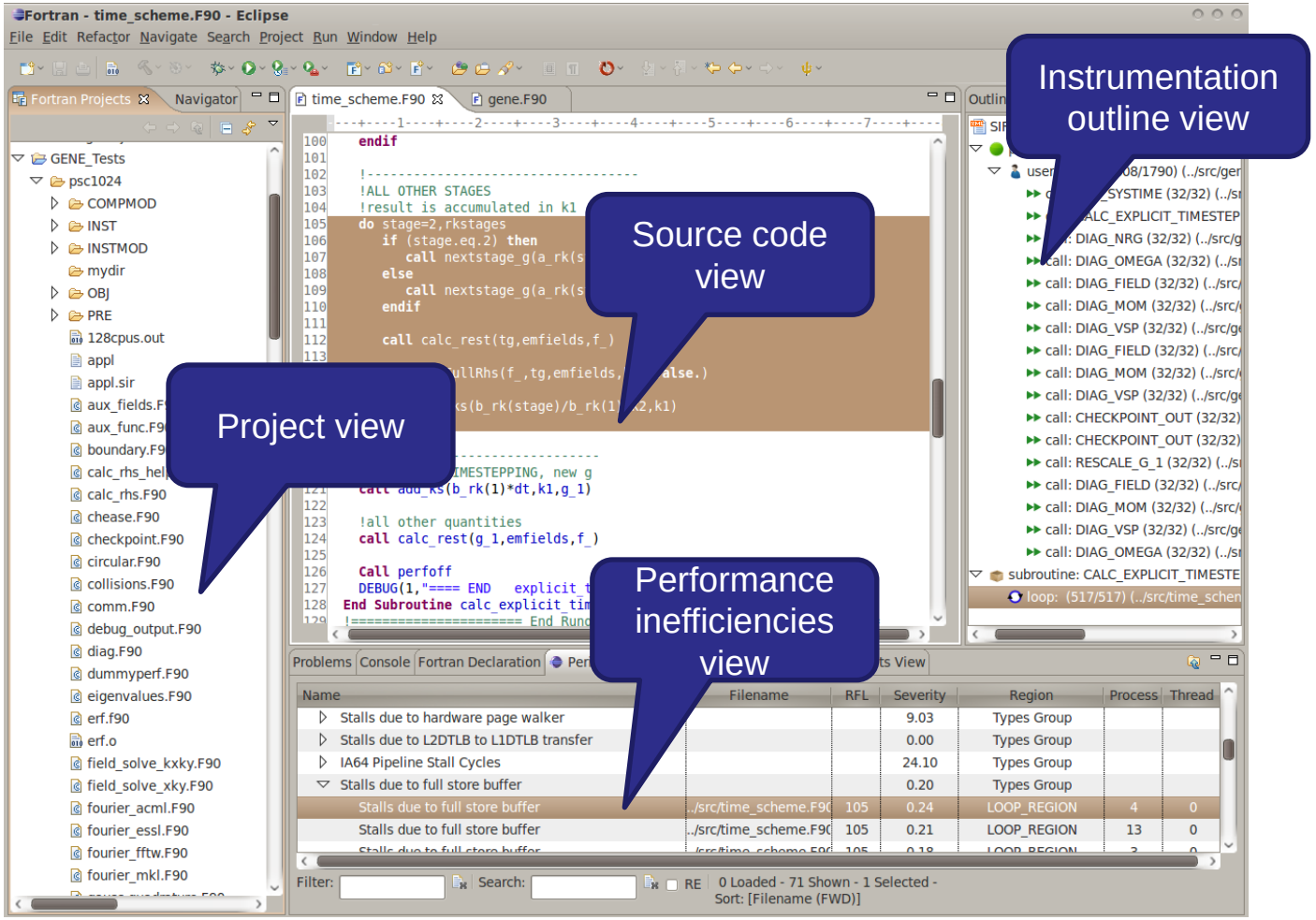
- Scalable distributed tree-like architecture
- Iterative on-line profiling using Score-P
- Automatic search for performance inefficiencies:
  - MPI wait states
  - Single core stalls
  - OpenMP overheads
  - OpenMP scalability
- New BSD license

http://www.lrr.in.tum.de/periscope



Periscope GUI

Eclipse-based integrated development and performance analysis environment



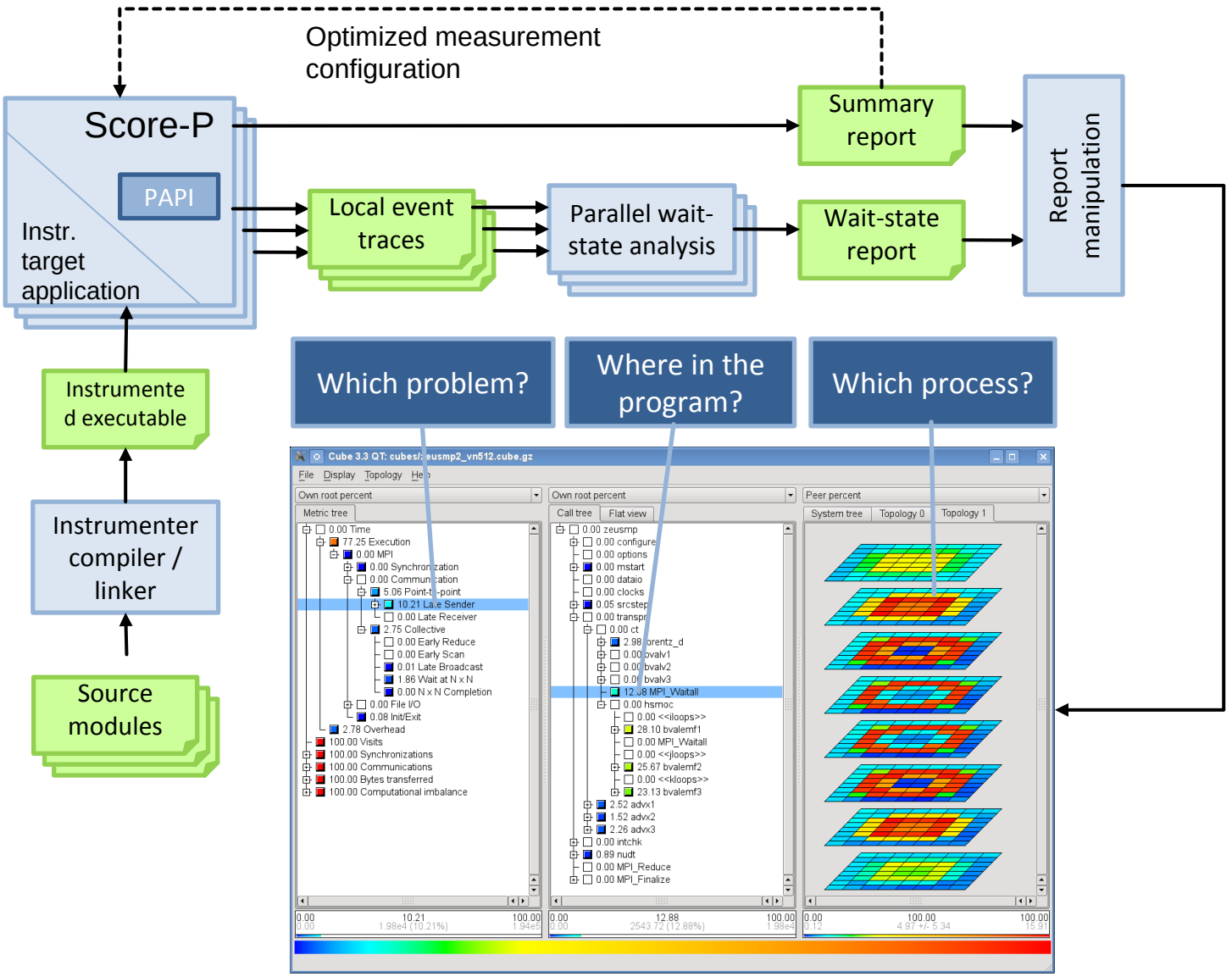
- Scalable performance-analysis toolset for parallel codes
  - Focus on communication & synchronization
- Integrated performance analysis process
  - Performance overview via call-path profiling
  - In-depth study of application behavior via event tracing
- Programming models
  - MPI, OpenMP
  - Future: support for PGAS and accelerators

JÜLICH FORSCHUNGSZENTRUM

German Research School for Simulation Sciences

www.scalasca.org

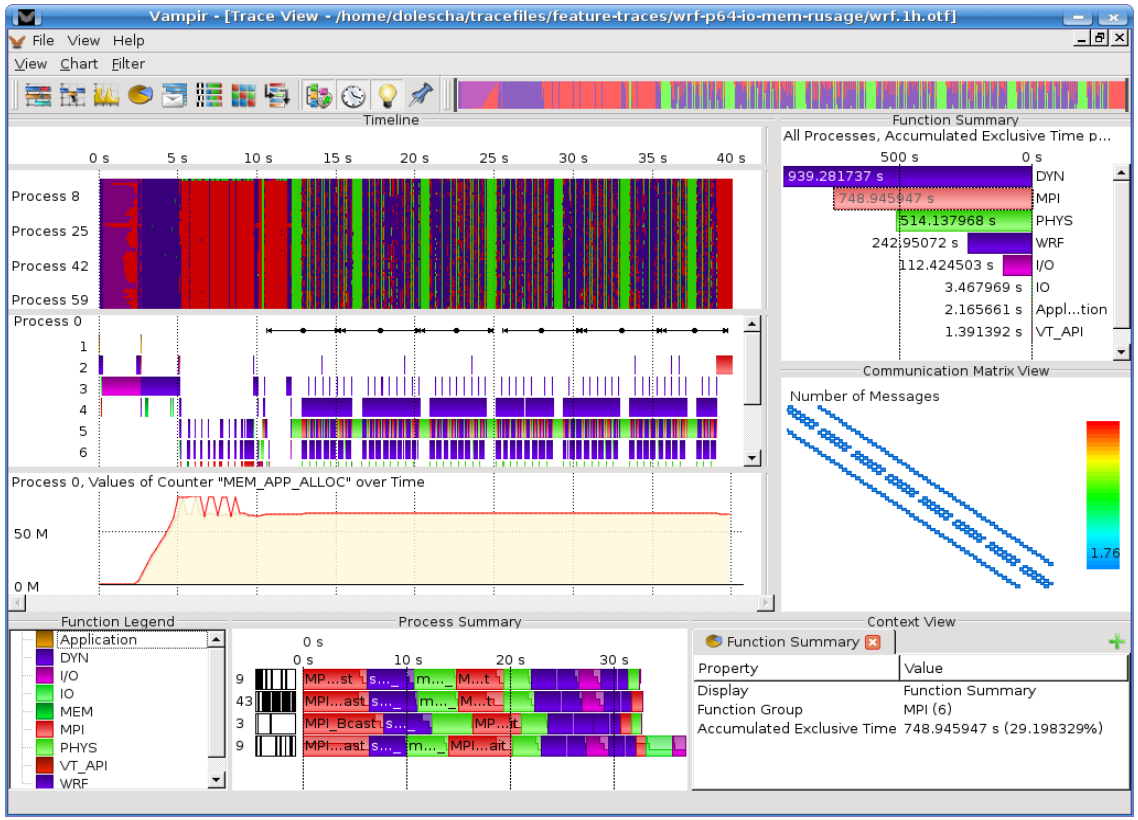
Scalasca architecture



Vampir – Interactive Performance Analysis

Highly scalable interactive performance analysis with event trace visualization

- Multiple displays:
- Global Timeline
- Process Timeline
- Performance Counters
- Communication
- Statistics
- and more



TECHNISCHE UNIVERSITÄT DRESDEN

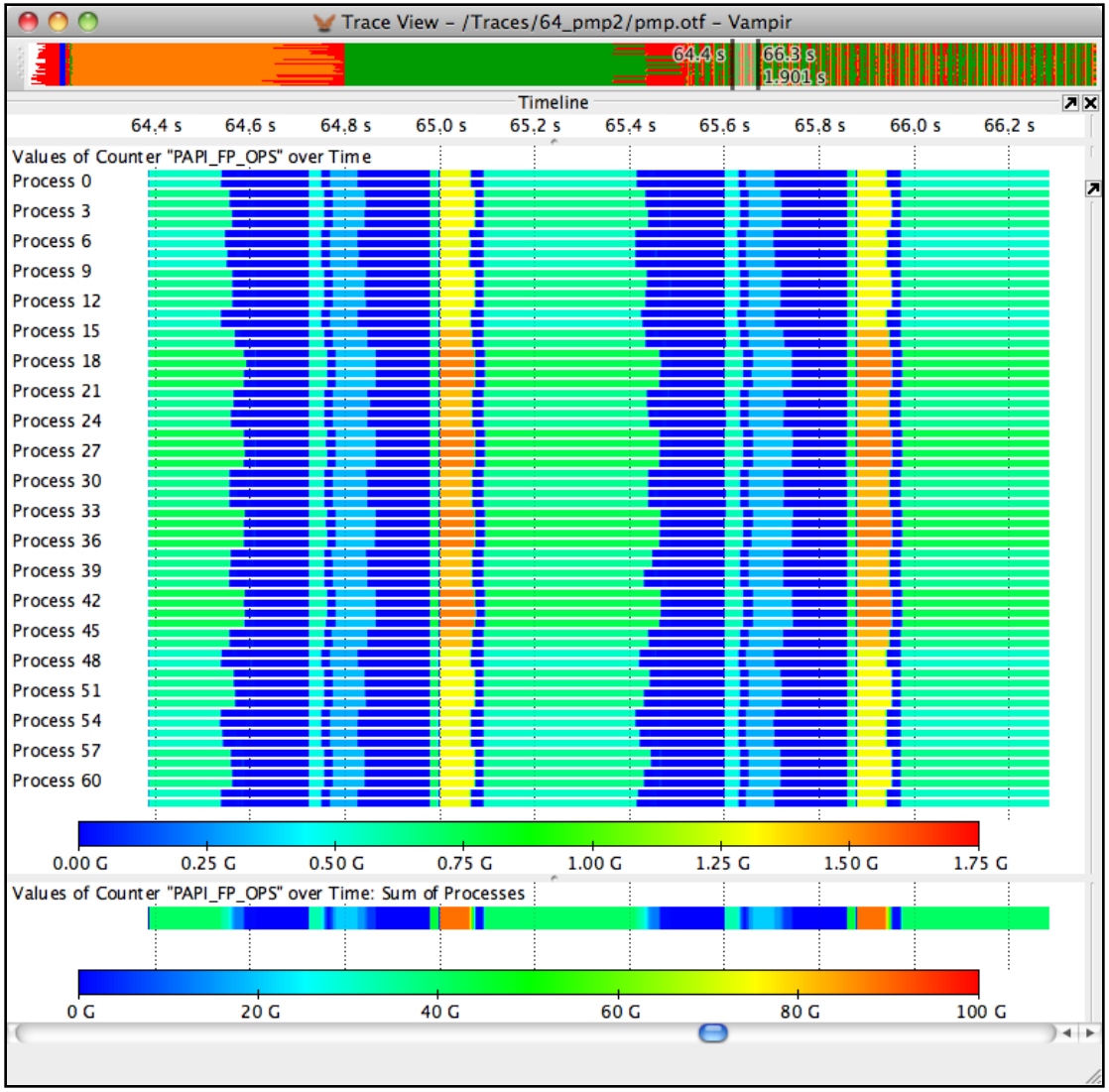
Vampir – Interactive Performance Analysis

Scalability of inter-active visualization:

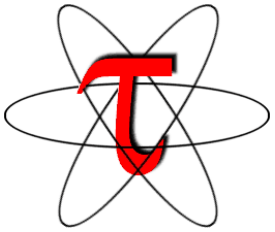
- VampirServer with parallel distributed performance data processing
- 100 000 processes and terabytes of performance data

Scalable displays:

- Process clustering
- Performance Radar



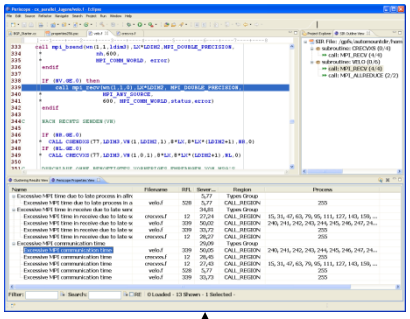
- Profiling and tracing toolkit with automatic instrumentation, measurement and analysis support
  - Uses Score-P for profiling and tracing using native OTF2 generation capability
  - Fortran, C++, C, UPC, Java, Python, Chapel
  - MPI, pthreads, OpenMP, CUDA, OpenCL, OpenSHMEM, and a combination of threads and MPI
  - Goal: to support all HPC platforms, compilers and runtime systems. Widely ported.
  - 3D profile browser, ParaProf, data mining and cross experiment analysis tool, PerfExplorer, and performance database technology (PerfDMF)



TAU Performance System® (continued)

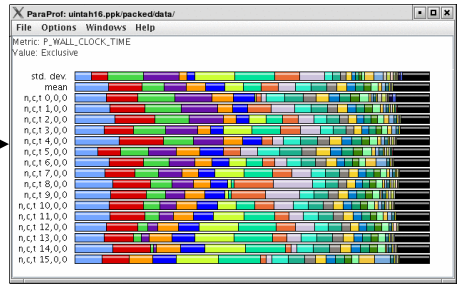
- Supports automatic instrumentation:
  - PDT: Source level (routines, loops, memory, I/O, UPC constructs)
  - Compiler based instrumentation (GNU, IBM, NAG, Intel, PGI, Pathscale, Cray)
  - Binary rewriter (MAQAO, DyinstAPI)
  - Automatic wrapper library generator (tau_gen_wrapper)
  - MPI wrapper, I/O, Memory wrapper library
- Supports callstack debugging, I/O and heap memory usage, leak detection
- Interfaces with Score-P, PAPI, Vampir, MAQAO, and Scalasca
- BSD style license
- http://tau.uoregon.edu

The Whole Ensemble



Periscope

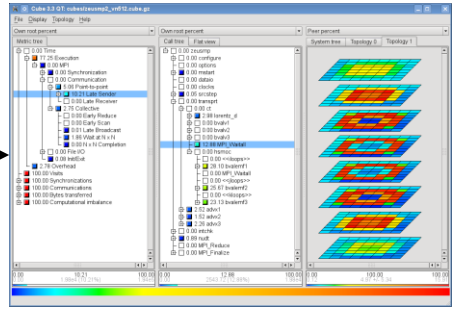
TAU
PerfExplorer



TAU Paraprof

CUBE4
report

CUBE4
report



CUBE

Online interface

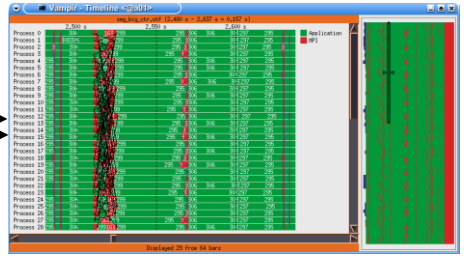
Score-P

PAPI

Instr.
target
application

Scalasca
wait-state
analysis

OTF2
traces



Vampir

Overview

- Introduction and Motivation *by Andreas Knüpfer*
- Performance Tools Overview *by Felix Wolf*
- **Score-P and Tools Demo** *by Sameer Shende*
- Conclusions and Discussion

Demo

Overview

- Introduction and Motivation *by Andreas Knüpfer*
- Performance Tools Overview *by Felix Wolf*
- Score-P and Tools Demo *by Sameer Shende*
- **Conclusions and Discussion**

Follow-up projects and future goals

- H4H (2010 – 2013)
  - Heterogeneous architectures
- HOPSA (EU-Russia project, 2011-2012)
  - Integration with system monitoring
- LMAC (2011 – 2014)
  - Evolution of Score-P
  - Performance dynamics
- GASPI (2011-2014)
  - PGAS library, performance API, tool support

- Open for new tools or groups in the future

Status

- Get pre-release version including demo cases
  - At webpage http://www.score-p.org
  - At VI-HPS/POINT LiveDVD
- Official 1.0 release in Dec. 2011
- Contacts:
  - Andreas Knüpfer andreas.knuepfer@tu-dresden.de
  - Felix Wolf f.wolf@grs-sim.de
  - Sameer Shende sameer@cs.uoregon.edu

Discussion

- How can we extend the Score-P community?
- In which way can a new feature be requested/provided?
- What future research directions are worthwhile?

…