KRELL
institute

Open | SpeedShop™

Open Source Performance Analysis for Linux SSI and Clusters

Paradyn Conference 2007

Jim Galarowicz, Krell Institute
Martin Schulz, LLNL

Trademark Acknowledgements

Intel, Intel Inside (logos) and Itanium are trademarks of Intel Corporation in the United States, other countries, or both.

Linux is a trademark of Linus Torvalds in the United States, other countries or both.

Qt and the Qt logo are trademarks of Trolltech in Norway, the United States and other countries.

SGI SpeedShop, IRIX, SGI and SGI Altix are trademarks of Silicon Graphics Inc.

IBM is a registered trademarks of International Business Machines Corporation in the United States, other countries, or both.

– *All other trademarks mentioned herein are the property of their respective owners*

Agenda

- What is Open|SpeedShop?
- Project Highlights/Overview
- Features
- Dyninst Usage in Open|SpeedShop
- Current Status
- Future Plans
- Questions

KRELL Institute

What is Open|SpeedShop?

- Comprehensive Open Source Parallel Performance Analysis Framework
  - **Goal:** *One* **tool for all performance analysis needs**
  - **Targets <u>Users</u>** *and* **<u>Tool Developers</u>**
- **Funding**
  - **DOE/NNSA as part of ASC PathForward**
  - **Initial phase co-funded by SGI**
- **Status**
  - **Version 1.0 available as source and RPMs**
  - **Source code is GPL/LGPL**

Partners

- Krell Institute
  - **Hosts Development**
- **DOE/NNSA Tri-Labs**
  - **Lawrence Livermore**
  - **Los Alamos**
  - **Sandia**
- **University of Wisconsin & University of Maryland**
  - **DynInst & Infrastructure**

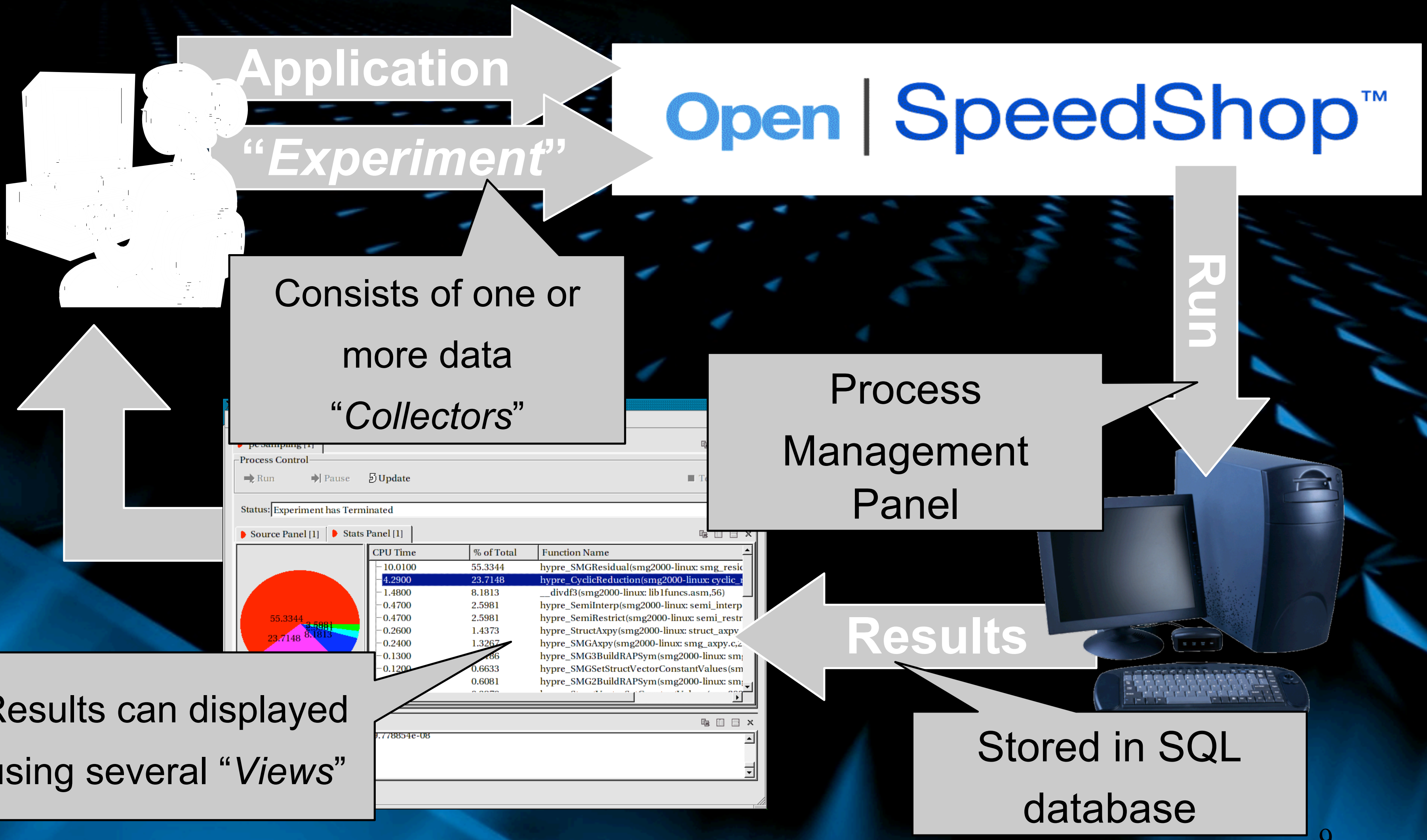
Overview / Highlights

- Open Source Performance Analysis Tool
  - **Most common performance analysis steps in** *all in one tool*
  - *Extensible* **by using plugins for data collection and viewing**
  - **GPL/LGPL license**
- **Instrumentation at Runtime**
  - **Use of** *unmodified application binaries*
  - *Attach/Detach to/from* **running executables/applications**
  - *Load and Start* **executables/applications into tool**
- **Flexible and Easy to use user interfaces**
  - *GUI* **with wizards to guide users through creation of experiment**
  - *Command Line*
  - *Batch*
  - *Python Scripting*

Overview / Highlights

- **Large Range of Platforms**
  - *Linux Clusters* **with x86, IA-64, Opteron, and EM64T CPUs**
  - **SGI** *SSI* **systems**
  - **Designed with** *portability* **in mind**
- **Availability**
  - **Used at** *all three ASC labs* **with lab-size applications**
  - **Source and RPM versions available**
  - *www.openspeedshop.org*
- Linux versions
  - Tested on typical Linux distributions (including *SLES, RHEL, Fedora Core, Suse ....*)

Typical Workflow

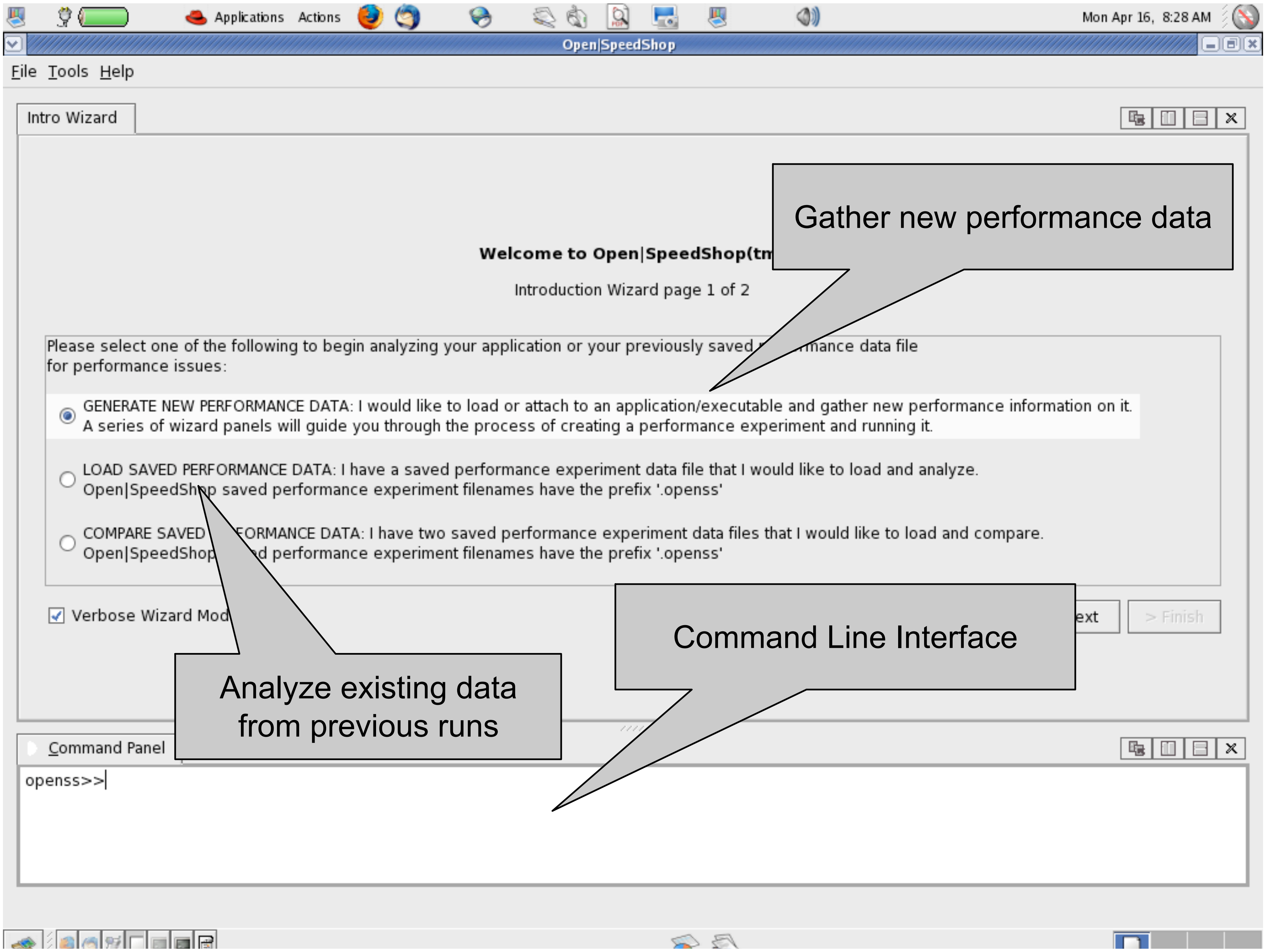


Status: Experiment has Terminated

CPU Time	% of Total	Function Name
10.0100	55.3344	hypre_SMGResidual(smg2000-linux: smg_resic
4.2900	23.7148	hypre_CyclicReduction(smg2000-linux: cyclic_r
1.4800	8.1813	__divdf3(smg2000-linux: lib1funcs.asm,56)
0.4700	2.5981	hypre_SemiInterp(smg2000-linux: semi_interp
0.4700	2.5981	hypre_SemiRestrict(smg2000-linux: semi_restr
0.2600	1.4373	hypre_StructAxpy(smg2000-linux: struct_axpy
0.2400	1.3267	hypre_SMGAxpy(smg2000-linux: smg_axpy.c,2
0.1300	[illegible]	hypre_SMG3BuildRAPSym(smg2000-linux: sm
0.1200	0.6633	hypre_SMGSetStructVectorConstantValues(sm
[illegible]	0.6081	hypre_SMG2BuildRAPSym(smg2000-linux: sm

Features: Performance Experiments

- Available Now:
  - PC sampling (*pcsamp*)
  - User time (*usertime*)
  - Hardware counter (*hwc, hwctime*)
  - MPI call tracing (*mpi, mpit*)
  - I/O call tracing (*io, iot*)
  - Floating Point Exception (FPE) tracing (*fpe*)
- Extensible
  - Plugin concept for collectors and views
  - Well defined/documented APIs



Welcome to Open|SpeedShop(tm

Introduction Wizard page 1 of 2

Please select one of the following to begin analyzing your application or your previously saved performance data file for performance issues:

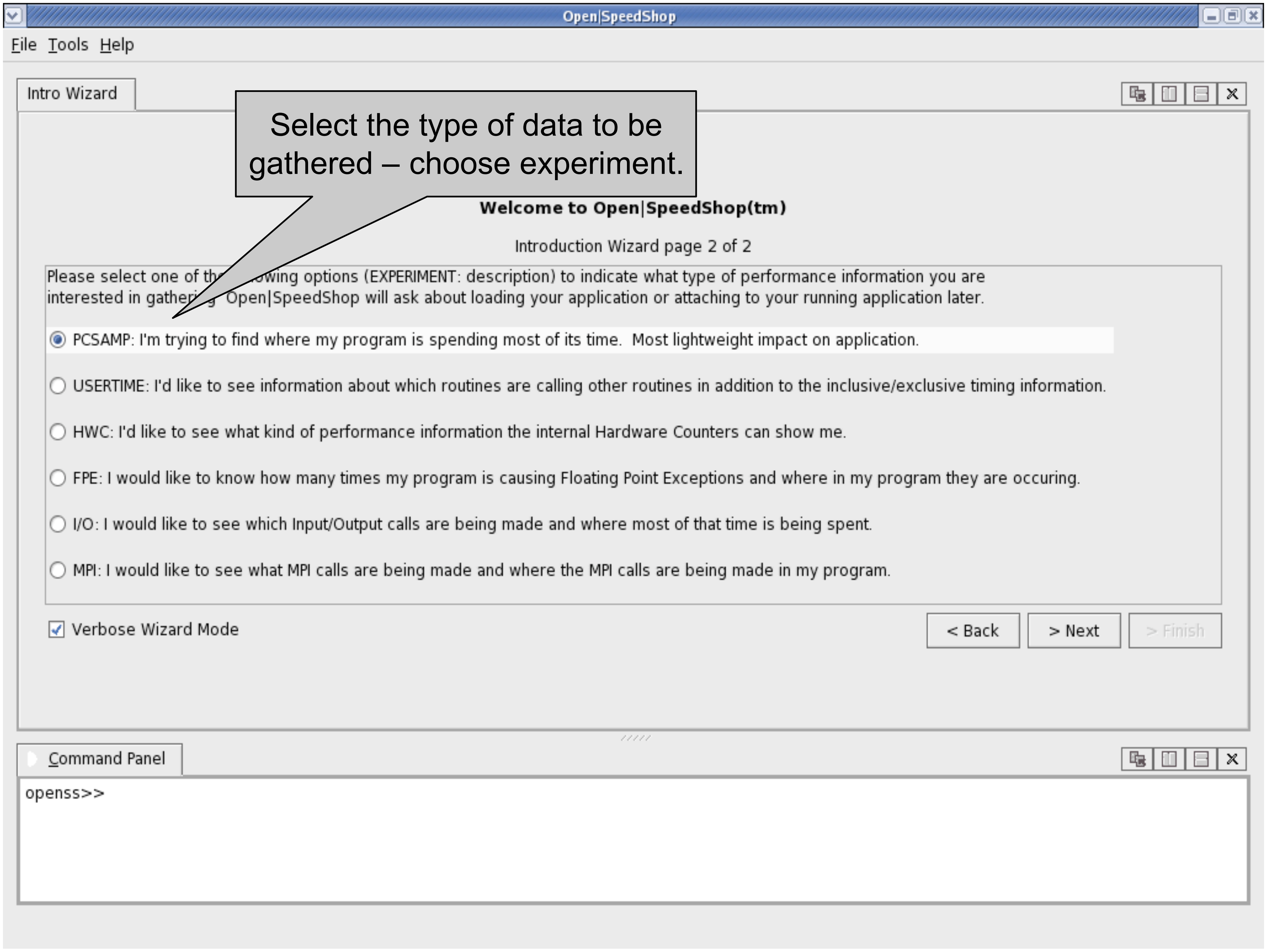
[✓ GENERATE NEW PERFORMANCE DATA: I would like to load or attach to an application/executable and gather new performance information on it. A series of wizard panels will guide you through the process of creating a performance experiment and running it.]

LOAD SAVED PERFORMANCE DATA: I have a saved performance experiment data file that I would like to load and analyze. Open|SpeedShop saved performance experiment filenames have the prefix '.openss'

COMPARE SAVED PERFORMANCE DATA: I have two saved performance experiment data files that I would like to load and compare. Open|SpeedShop saved performance experiment filenames have the prefix '.openss'

```
openss>>
```

Select the type of data to be gathered – choose experiment.



Welcome to Open|SpeedShop(tm)

Introduction Wizard page 2 of 2

Please select one of the following options (EXPERIMENT: description) to indicate what type of performance information you are interested in gathering. Open|SpeedShop will ask about loading your application or attaching to your running application later.

[✓ PCSAMP: I'm trying to find where my program is spending most of its time. Most lightweight impact on application.]

USERTIME: I'd like to see information about which routines are calling other routines in addition to the inclusive/exclusive timing information.

HWC: I'd like to see what kind of performance information the internal Hardware Counters can show me.

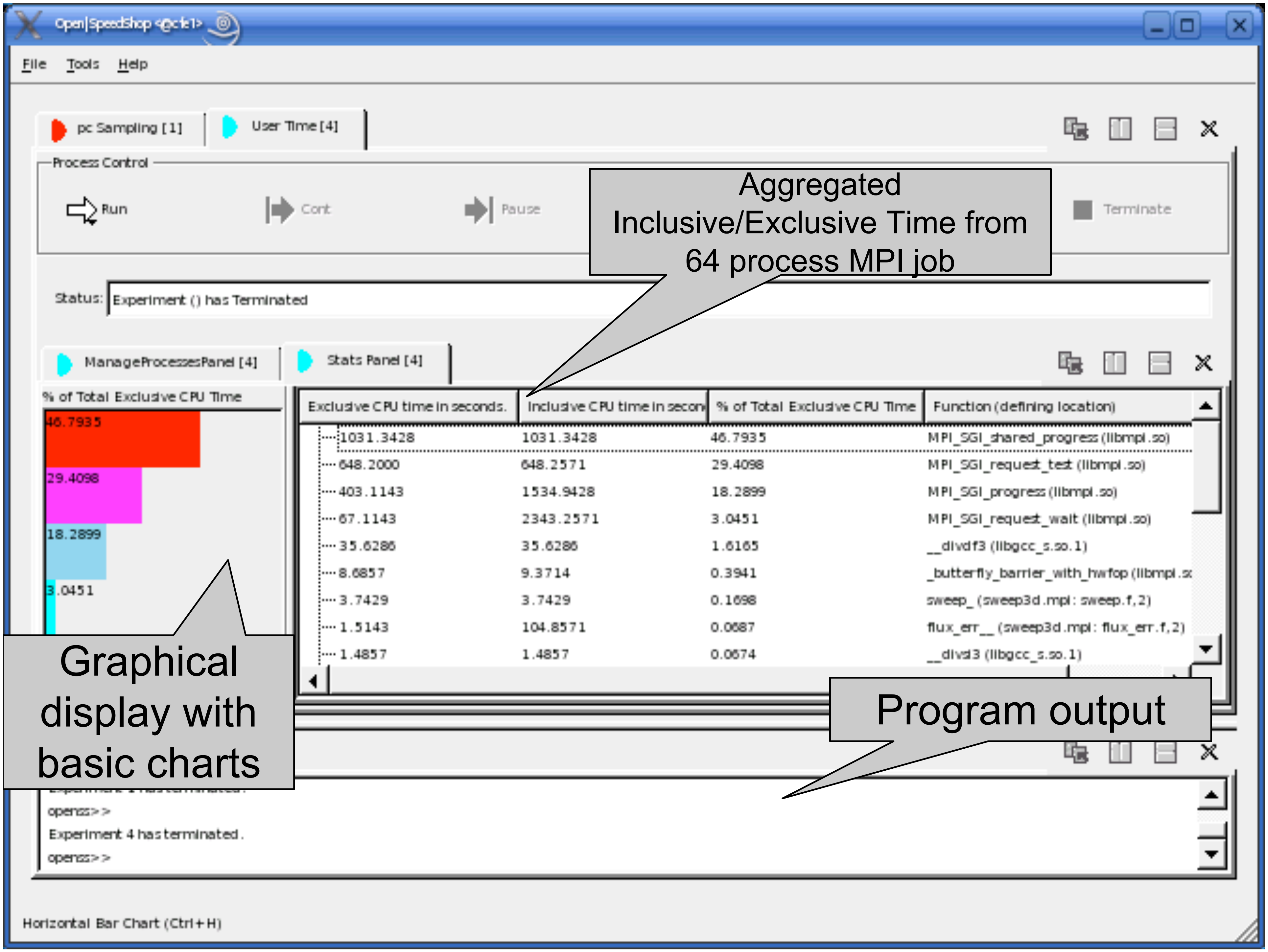
FPE: I would like to know how many times my program is causing Floating Point Exceptions and where in my program they are occuring.

I/O: I would like to see which Input/Output calls are being made and where most of that time is being spent.

MPI: I would like to see what MPI calls are being made and where the MPI calls are being made in my program.

[✓ Verbose Wizard Mode]

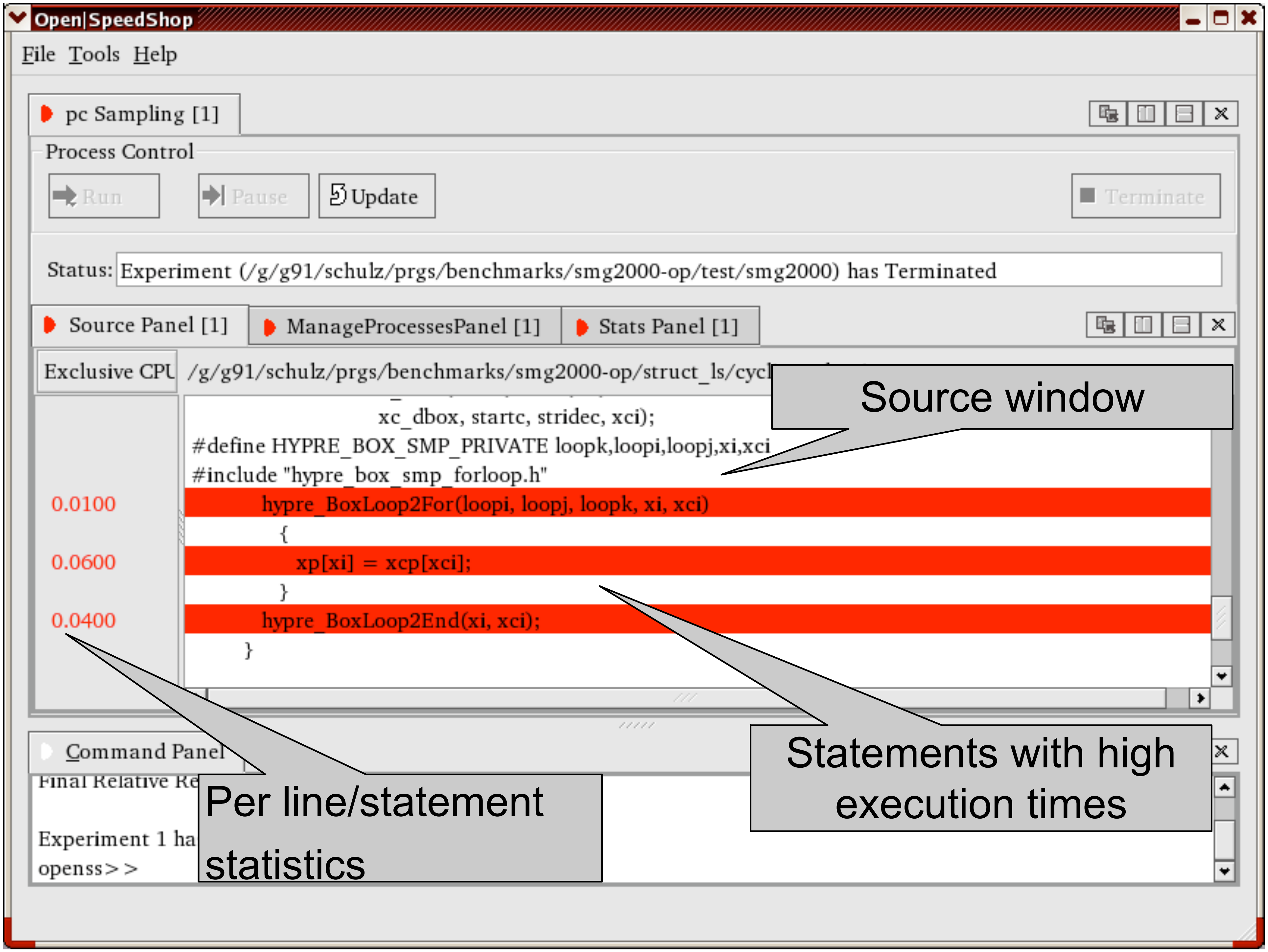
```
openss>>
```



Status: Experiment () has Terminated

Exclusive CPU time in seconds.	Inclusive CPU time in secon	% of Total Exclusive CPU Time	Function (defining location)
1031.3428	1031.3428	46.7935	MPI_SGI_shared_progress (libmpi.so)
648.2000	648.2571	29.4098	MPI_SGI_request_test (libmpi.so)
403.1143	1534.9428	18.2899	MPI_SGI_progress (libmpi.so)
67.1143	2343.2571	3.0451	MPI_SGI_request_wait (libmpi.so)
35.6286	35.6286	1.6165	__divdf3 (libgcc_s.so.1)
8.6857	9.3714	0.3941	_butterfly_barrier_with_hwfop (libmpi.s
3.7429	3.7429	0.1698	sweep_ (sweep3d.mpi: sweep.f,2)
1.5143	104.8571	0.0687	flux_err__ (sweep3d.mpi: flux_err.f,2)
1.4857	1.4857	0.0674	__divsi3 (libgcc_s.so.1)

```
openss>>
Experiment 4 has terminated.
openss>>
```



Status: Experiment (/g/g91/schulz/prgs/benchmarks/smg2000-op/test/smg2000) has Terminated

/g/g91/schulz/prgs/benchmarks/smg2000-op/struct_ls/cycl

```
                         xc_dbox, startc, stridec, xci);
        #define HYPRE_BOX_SMP_PRIVATE loopk,loopi,loopj,xi,xci
        #include "hypre_box_smp_forloop.h"
0.0100           hypre_BoxLoop2For(loopi, loopj, loopk, xi, xci)
                 {
0.0600              xp[xi] = xcp[xci];
                 }
0.0400           hypre_BoxLoop2End(xi, xci);
              }
```

```
Final Relative Re
Experiment 1 ha
openss>>
```

Parallel Performance Analysis

- Open|SpeedShop supports MPI and Multithreading
  - MPI Process control using MPIR interface
  - Works with multiple MPI implementations
  - Currently: *mpich, openmpi, lampi, lam, slurm, mpt*
  - Attach to running appl. or create appl. within O|SS
- Parallel Experiments
  - Apply sequential collectors to all nodes
  - Specialized MPI tracing experiments
- Results
  - By default results are aggregated
  - Optional: select individual processes
  - Compare or group ranks

Advanced Capabilities

- Stack trace views
  - Included in tracing and user time experiments
  - Visualize as call-tree and trace-back
- Experiment and Rank/Process/Thread Comparisons
- View results by Time segments
- Multi-rank analysis
  - Restrict results to task sets
  - Compare tasks or task sets
  - Cluster Analysis (grouping similar processes)

Dyninst in Open|SpeedShop

- Obtain and Process Application Symbols
- Attach to a running process
- Insert Code into Application Dynamically
  - Execute at Entry and Exit
  - Execute Now
  - Execute In Place of
- Control the Process/Application (start, stop, ...

Current Status: Open|SpeedShop

- Project being funded by NNSA/DOE
  - Two full-time developers and one part time
  - Developing new features
  - Bug fixing and support
- Project is on sourceforge
  - Can download source and rpms
  - Submit bug reports, comments, requests
  - Version 1.00 Released last November (SC07)
- ASC labs (LLNL, LANL, Sandia) are main users
- Several other users are in contact with team

Future plans: Open|SpeedShop

- Target: Peta-Scale machines
  - Data Collection and Transport
  - Result storage, aggregation, and analysis
- Offline Collectors
  - Execute experiments without tool backend
  - Targets microkernel architectures
- Fully disconnect GUI from framework
  - Remote execution with local GUI
  - Built on Command Line Interface (CLI)
- Long Term Vision
  - Performance "Cookbooks"
  - Help users plan experiments

Questions?

Jim Galarowicz
jeg@krellinst.org

Krell Institute
http://www.krellinst.org