TECHNISCHE UNIVERSITÄT DRESDEN

Center for Information Services and High Performance Computing (ZIH)

Introducing

OTF / Vampir / VampirTrace

Zellescher Weg 12

Willers-Bau A115

Tel. +49 351 - 463 - 34049

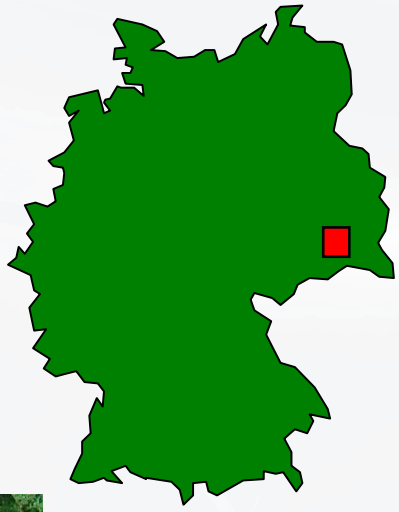
Robert Henschel (Robert.Henschel@zih.tu-dresden.de)

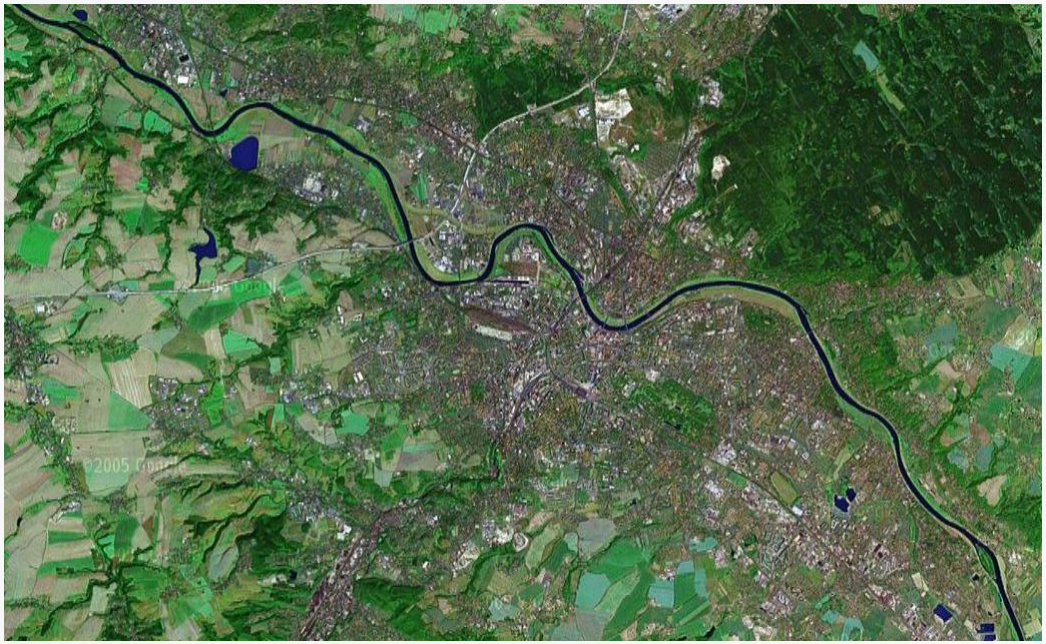
ZIH

Center for Information Services & High Performance Computing

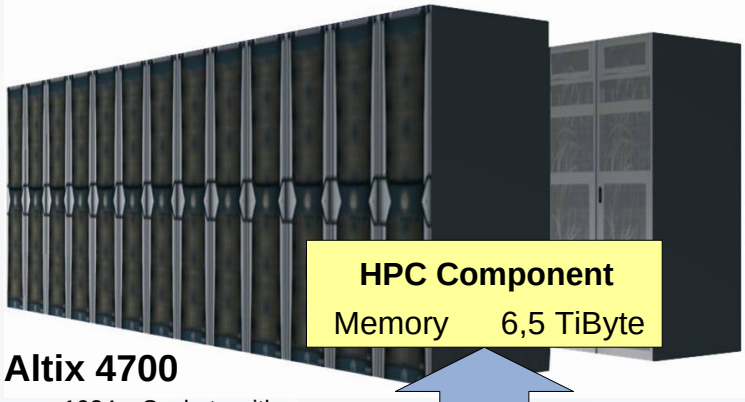
Dresden

- Infineon/Qimonda, AMD, etc.
- Public and private funded research institutes
- Evolving field: Molecular Biology and Bioinformatics
- TU Dresden
  - Center for Information Services and High Performance Computing (ZIH)





HPC Complex



HPC Component
Memory 6,5 TiByte

SGI Altix 4700

1024 x Sockets with Itanium2 Montecito Dual-Core CPUs (1.6 GHz/9MB L3 Cache)

8 GiB/s

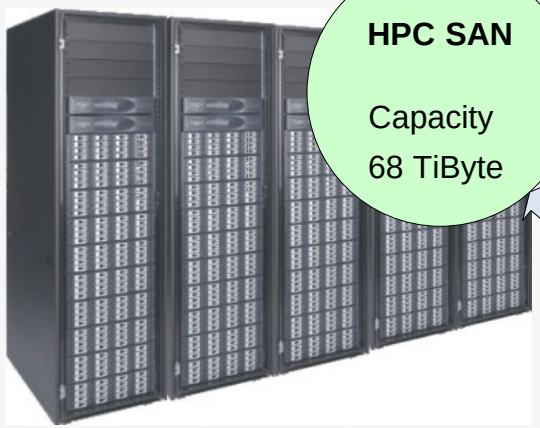
PC Farm

4 GiB/s

4 GiB/s

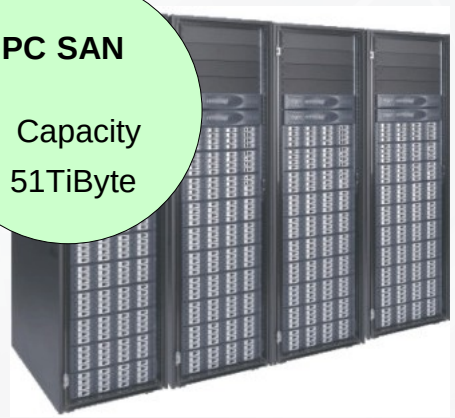
LNXI PC-Farm

AMD Opteron X85 Dual Core Chip with 2,6 GHz
384x Single CPU Nodes
232x Dual CPU Nodes
112x Quad CPU Nodes



HPC SAN

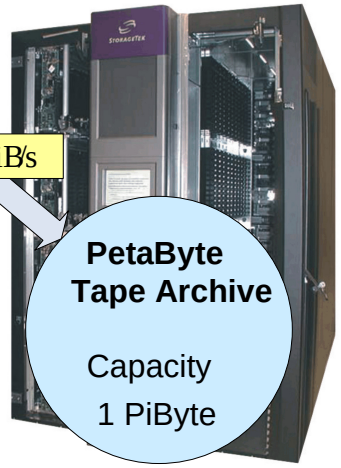
Capacity
68 TiByte



PC SAN

Capacity
51TiByte

1,8 GiB/s



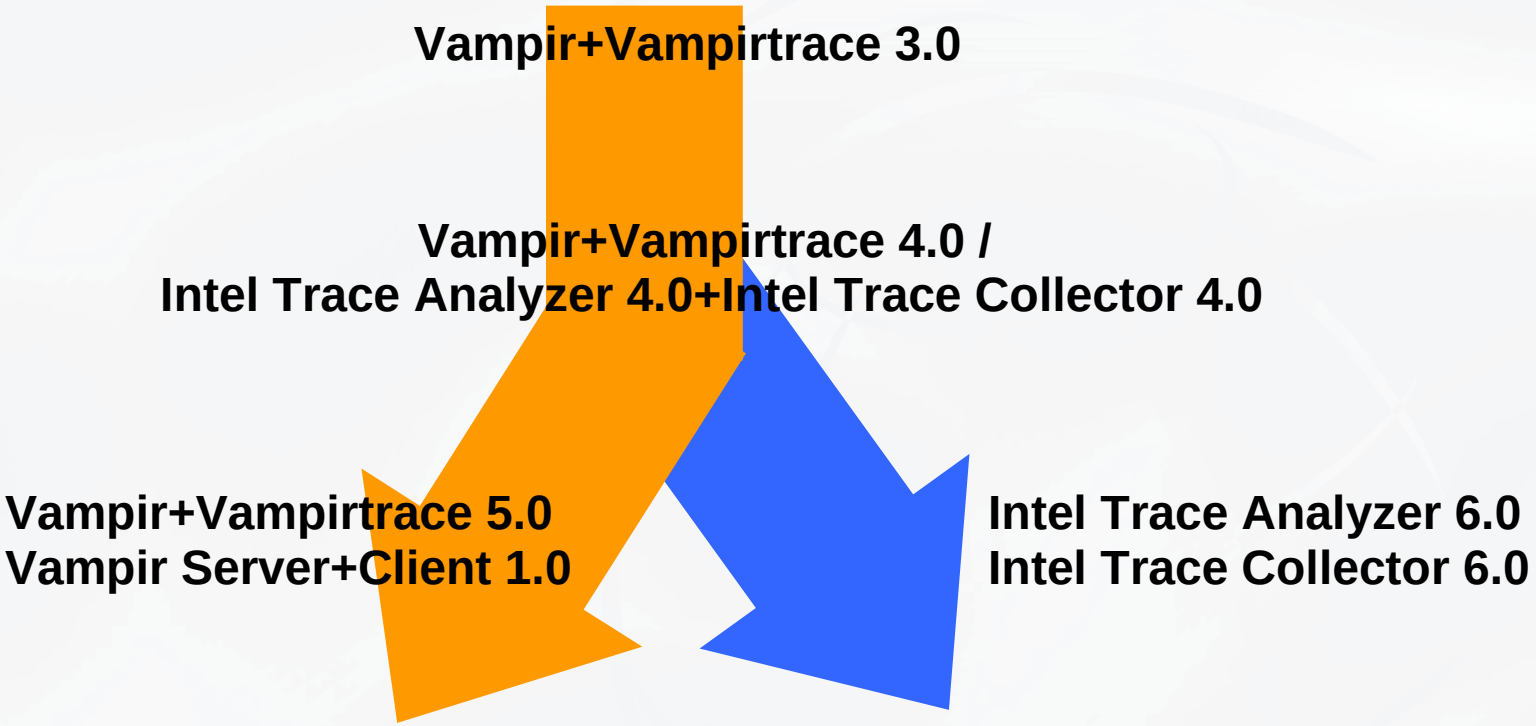
PetaByte Tape Archive

Capacity
1 PiByte

Tools Overview

- Tracing:
  - Collection of all events of a process / program
  - Sorted by time stamp
  - **VampirTrace**
- Trace File Format:
  - Fast and efficient sequential and parallel access
  - Platform independent
  - **OTF**
- Trace Visualization
  - Parallel/distributed server
  - Lightweight client on local workstation
  - **VampirServer and Client**

History


Vampir+Vampirtrace 3.0
Vampir+Vampirtrace 4.0 /
Intel Trace Analyzer 4.0+Intel Trace Collector 4.0
Vampir+Vampirtrace 5.0
Vampir Server+Client 1.0
Intel Trace Analyzer 6.0
Intel Trace Collector 6.0

Tracing: OTF Trace Format

- Open source trace file format
  - Available from the homepage of TU Dresden, ZIH
  - Google for *tu dresden* and *otf*
- Includes powerful *libotf* for use in custom applications
- Actively developed
  - In cooperation with the University of Oregon and Lawrence Livermore National Laboratory

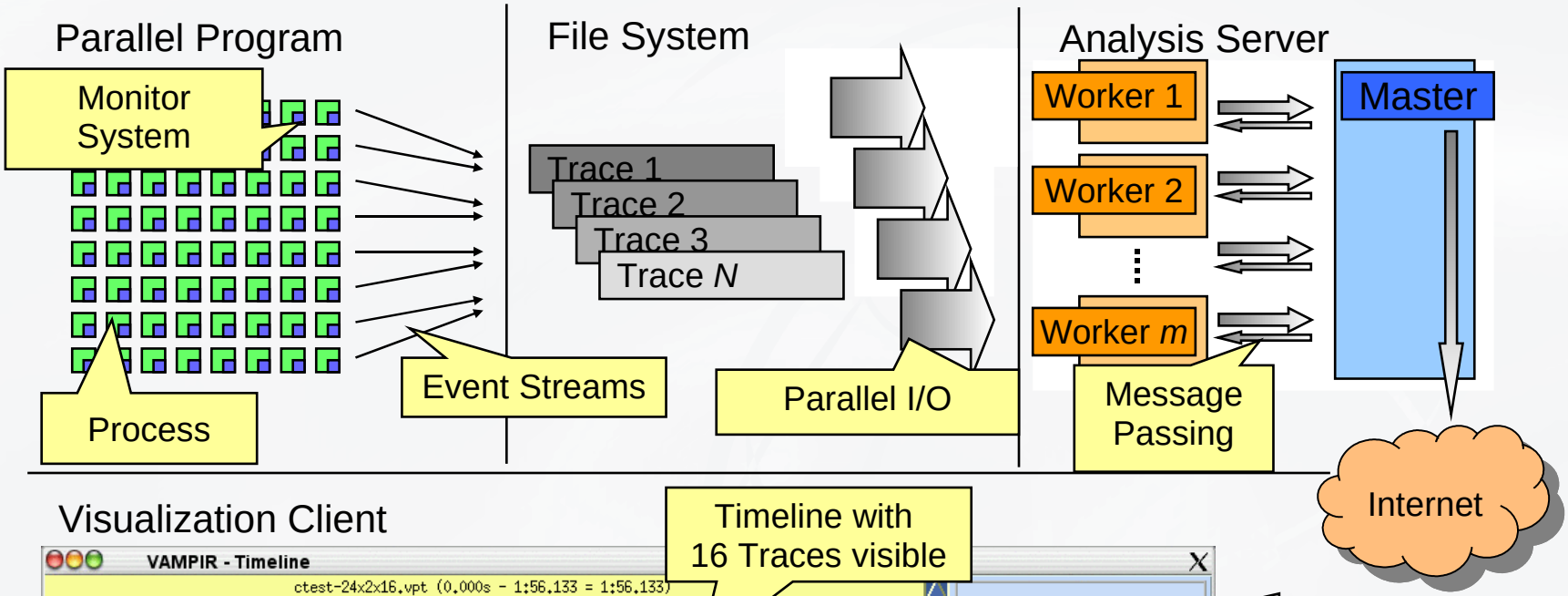
Tracing: VampirTrace

- Open source trace library
  - Available from the homepage of TU Dresden, ZIH
  - Google for *tu dresden* and *vampirtrace*
- Record events from applications
  - Function enter/leave
  - Process creation
  - MPI and OpenMP events
  - Hardware performance counters (PAPI)
- Collect event properties
  - Time stamp
  - Location (process / thread / MPI)
  - MPI specifics like message size etc.

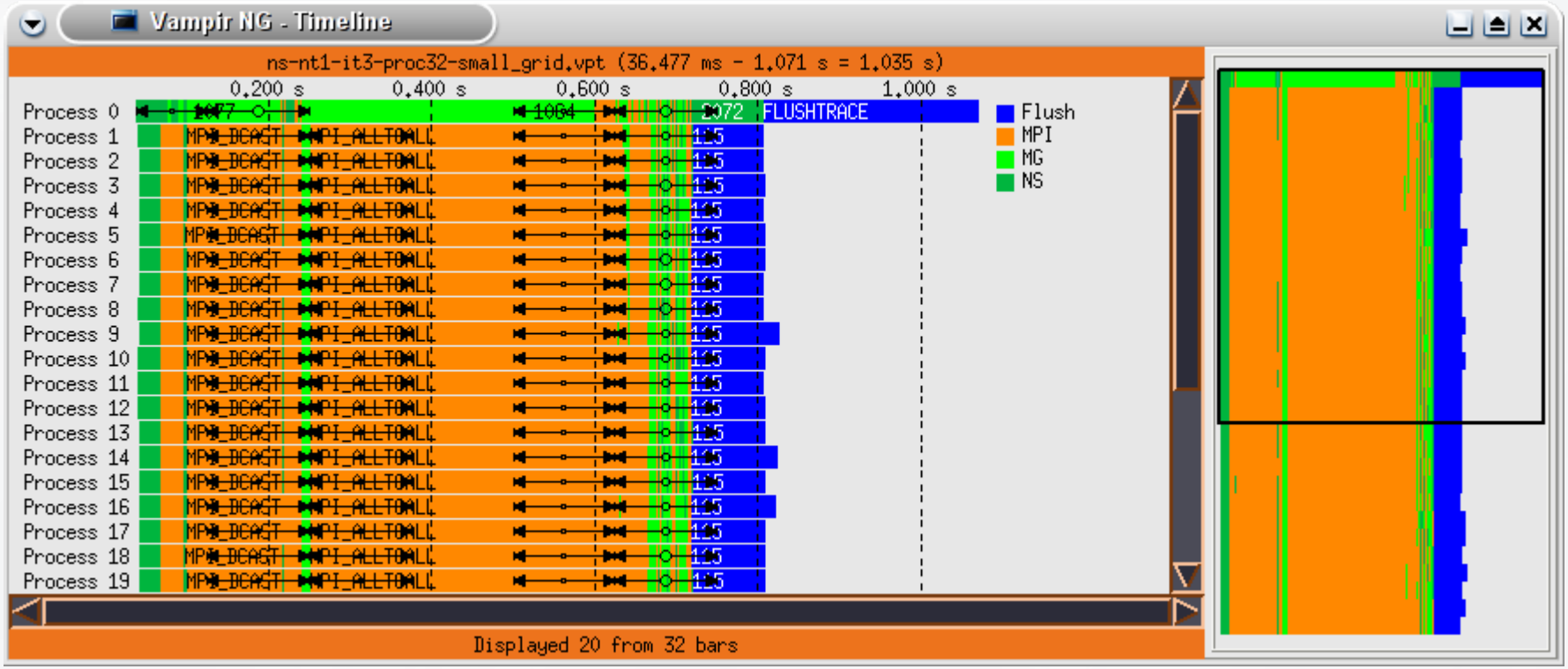
Trace Visualization: VampirServer and Client

- Parallel/distributed server
  - Runs in (part of) production environment
  - No need to transfer huge traces
  - Parallel I/O
- Lightweight client on local workstation
  - Receive visual content only
  - Already adapted to display resolution
  - Moderate network load
  - Scales to traces >40 GB

Trace Visualization: VampirServer and Client

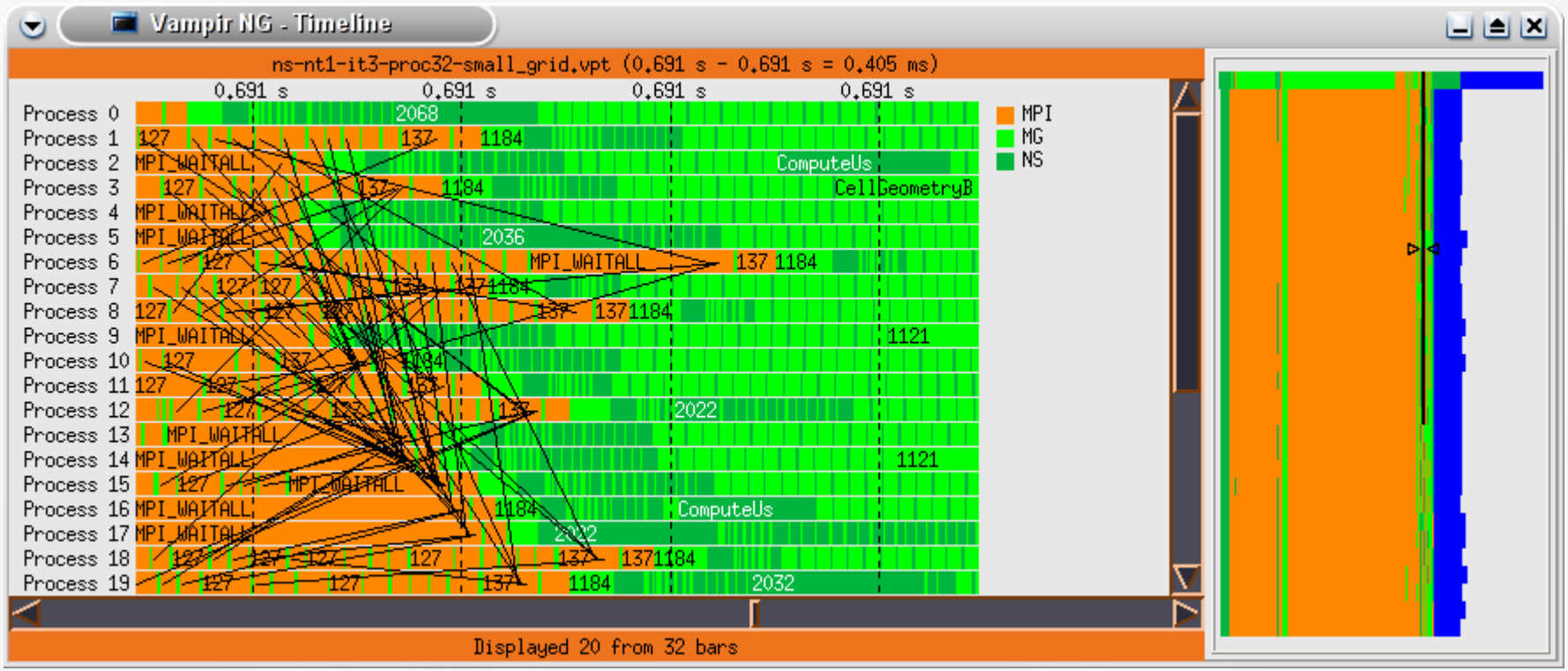


Trace Visualization: VampirServer and Client



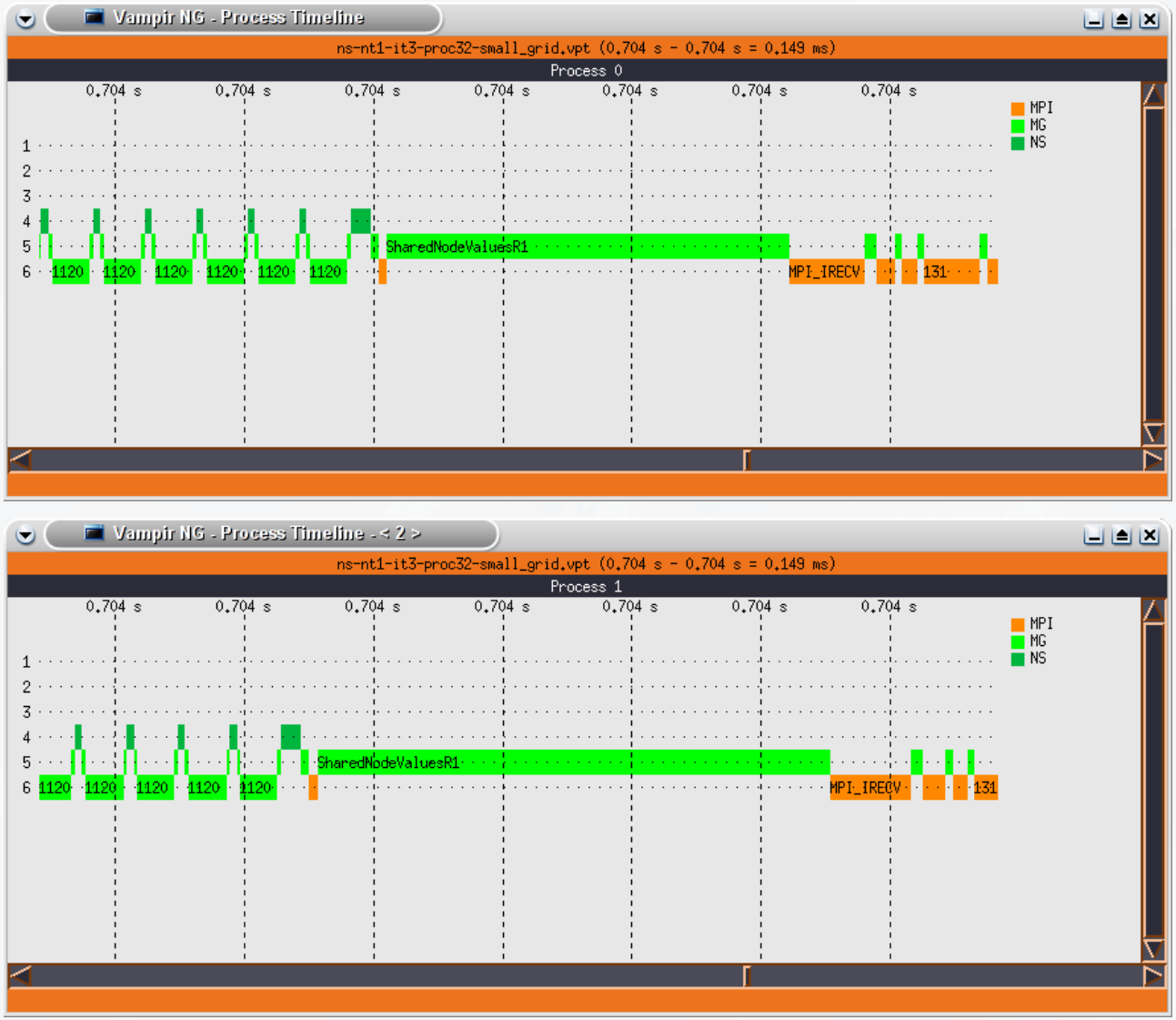
Global Timeline with Thumbnail View

Trace Visualization: VampirServer and Client



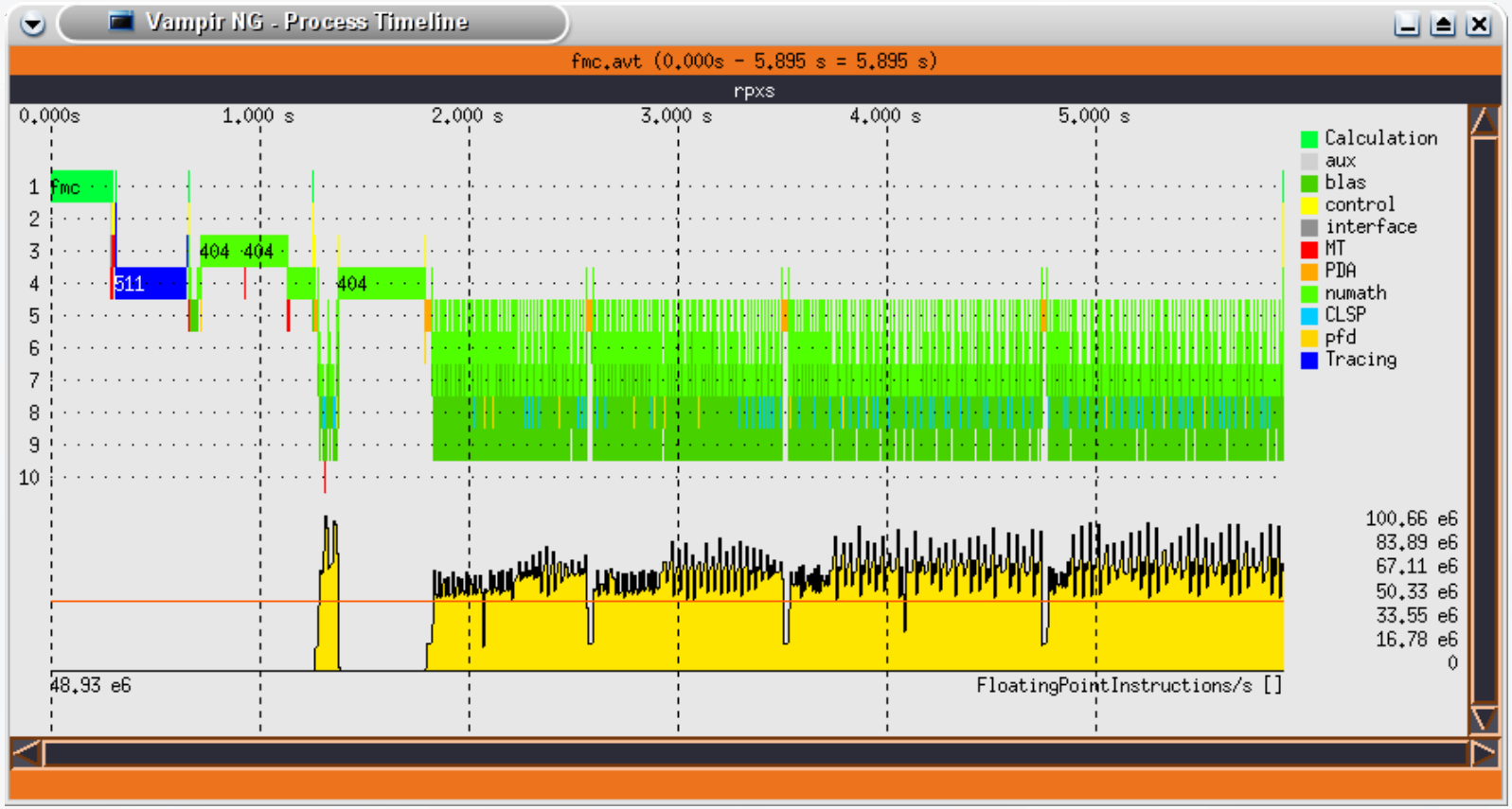
Zoomed

Trace Visualization: VampirServer and Client



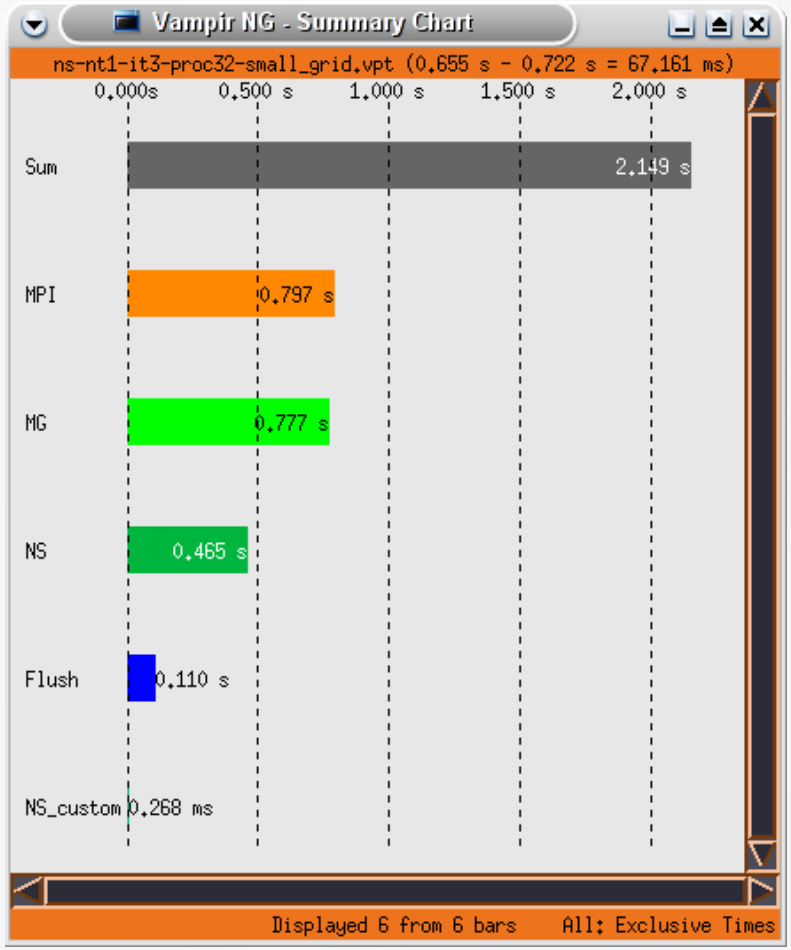
Aligned Process Timeline (Zoomed)

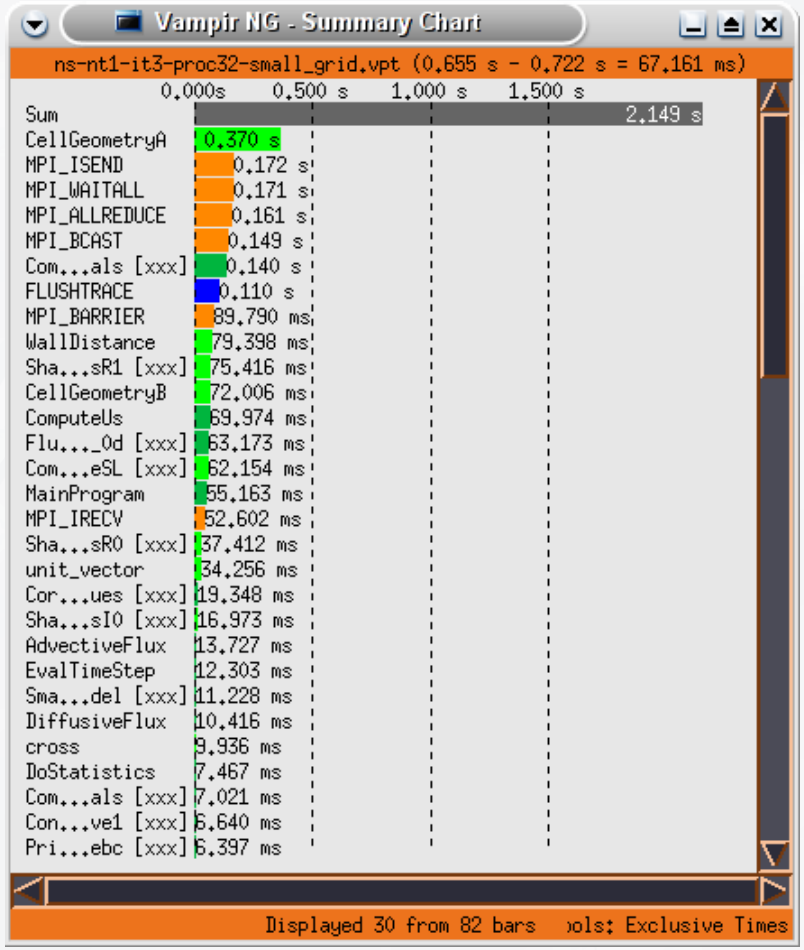
Trace Visualization: VampirServer and Client



Process Timeline with MFLOP Counter

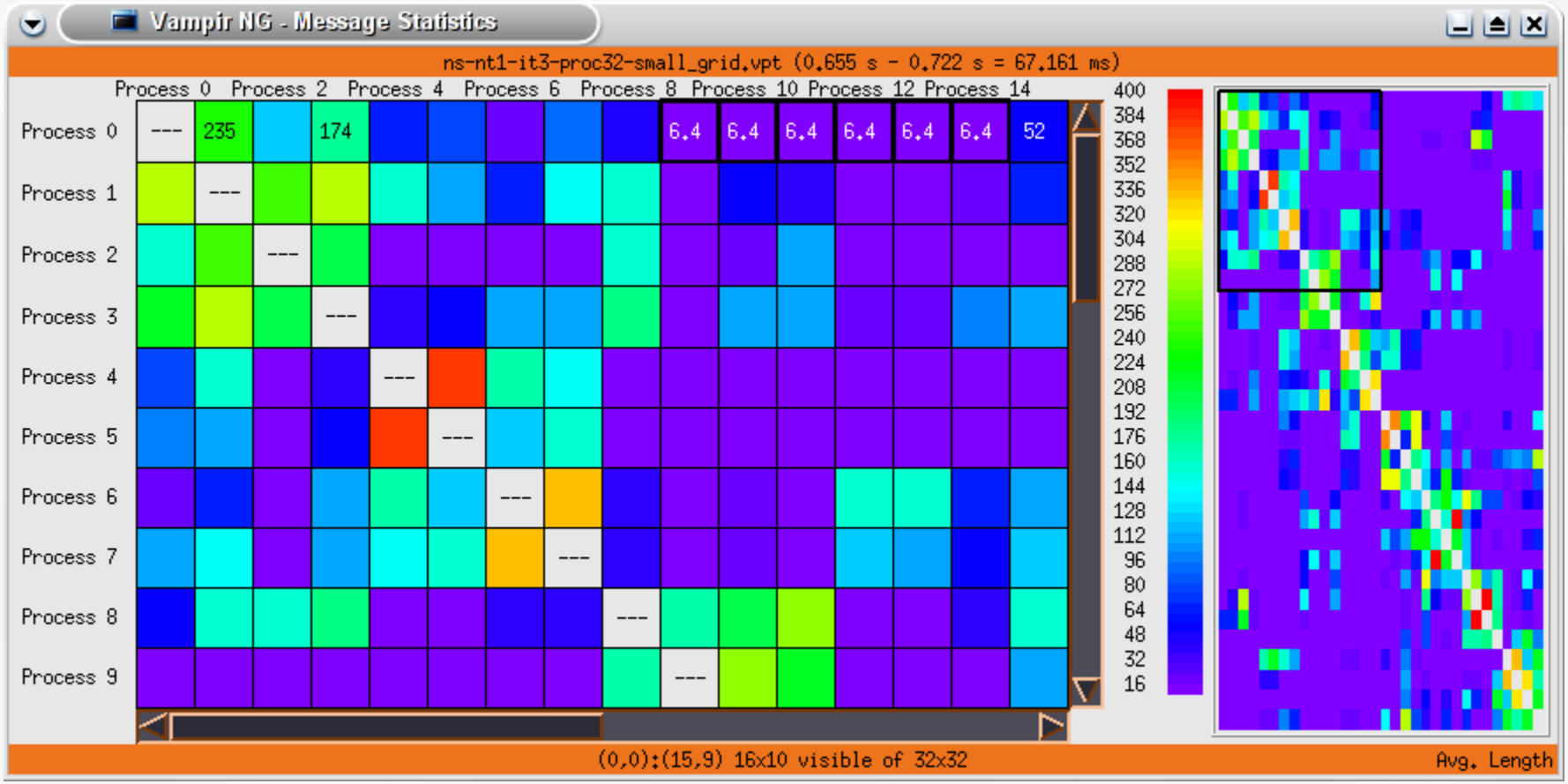
Trace Visualization: VampirServer and Client





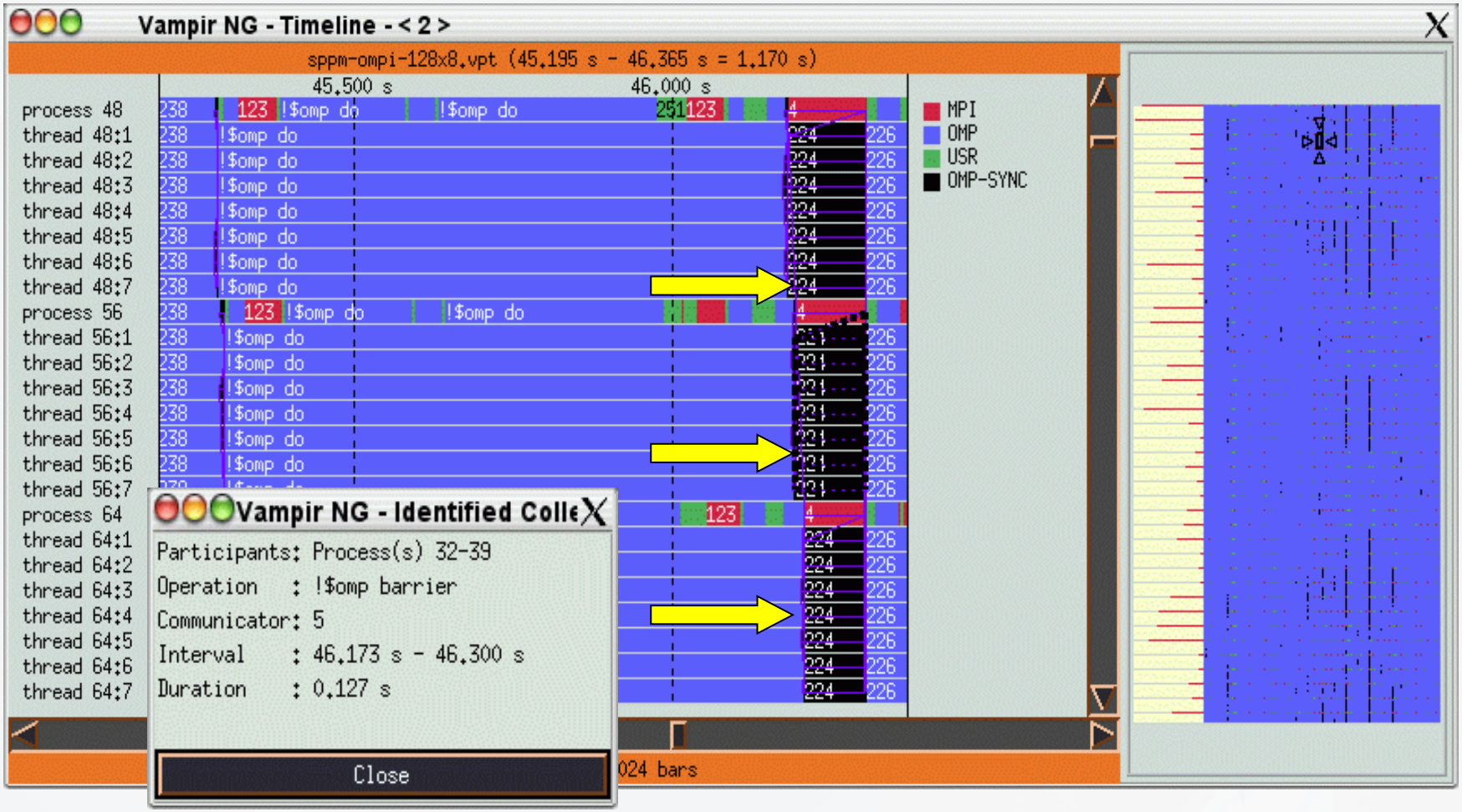
Grouped / Comprehensive Function Statistics

Trace Visualization: VampirServer and Client



Zoomed Message Statistics

Trace Visualization: VampirServer and Client



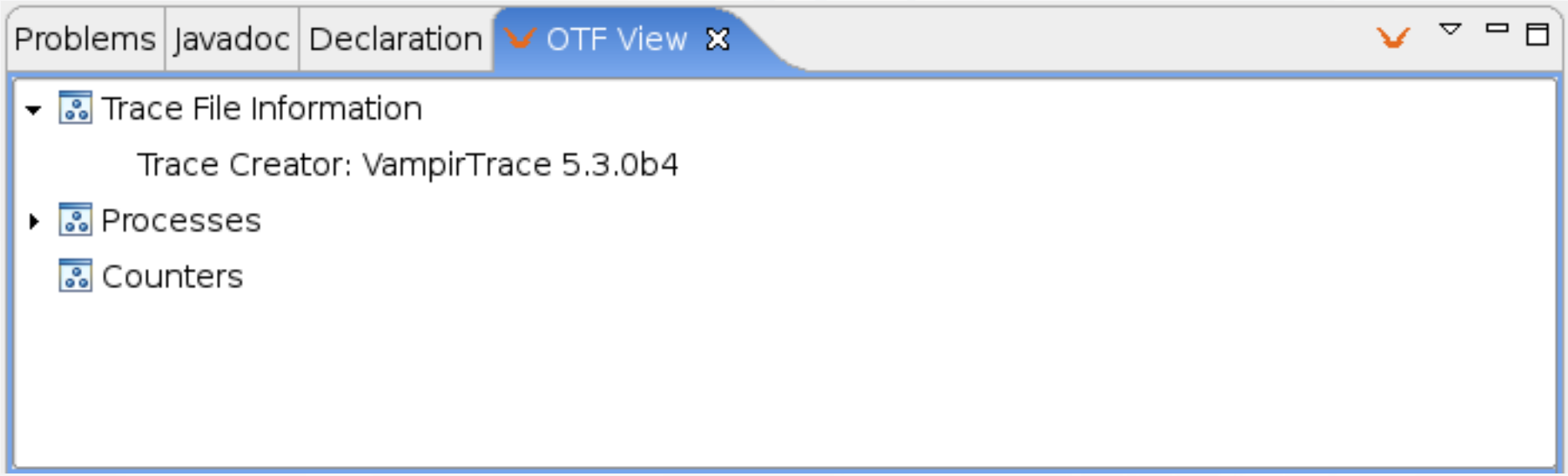
OpenMP Barrier Synchronization

Vampir and Eclipse

- We are currently evaluating how to integrate Vampir and Eclipse
  - What features of the Vampir GUI may be useful within Eclipse
  - What additional opportunities are created by having trace information and application source code within the same IDE
- Eclipse integration to better support application developers that are using performance analysis tools
  - Easy creation of trace files
  - Configuration of filters and function groups
  - Matching trace file information and source code location

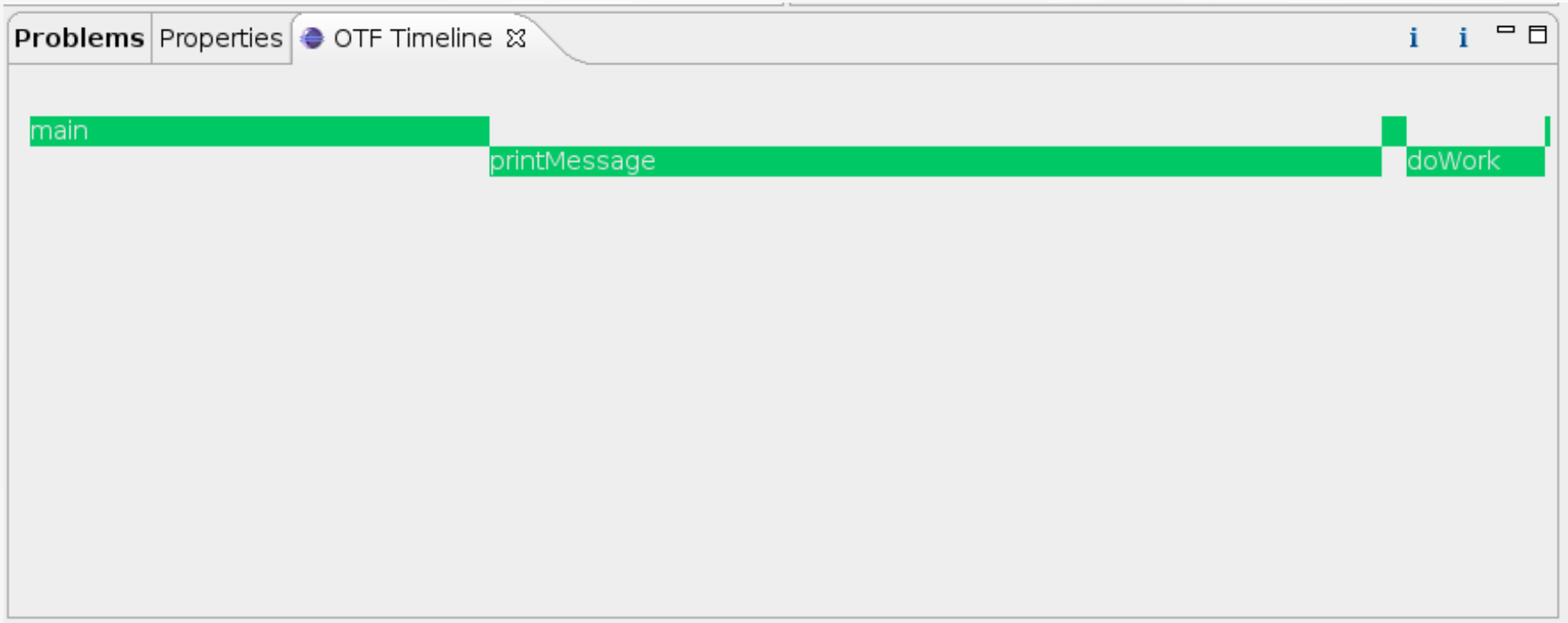
Vampir and Eclipse

- Displaying OTF trace file information in Eclipse
- Eclipse plug-in that displays OTF information and statistics
- JNI code to access *libotf* from Java applications
- Launching the Vampir GUI from within Eclipse



Vampir and Eclipse

- Test other views of Vampir inside Eclipse



Vampir and Eclipse

- VT_ macro awareness in Eclipse editor
  - Like MPI/OpenMP in PLDT
  - For example:
    - Check for matching VT_USER_START / VT_USER_END blocks
    - Create custom tracing blocks in the source code from selected areas
- Provide a wizard that will help the user to create a build target for tracing
- Provide a wizard to edit the launch configuration for a tracing run
  - Automatically set the VampirTrace specific environment variables
- Provide support to automatically build filters and groups
- Display information from OTF trace files in Eclipse (summary)
- Trace file management for projects
- Design a scaled down version of some Vampir displays for Eclipse
- Link those displays with the source code view of Eclipse

Thanks a lot for your attention!

Questions?