TECHNISCHE UNIVERSITÄT DRESDEN

DRESDEN concept

Martin Schroschk

Site Update - TU Dresden

Tools in HPC

May 15th, 2025 / NUG

ZIH
Information Services and
High Performance Computing

Agenda

1. Performance Engineering Tools Extendend for NEC SX-Aurora TSUBASA
2. Enforcing Data Lifecycle Managment

Performance Engineering Tools Extended for NEC SX-Aurora TSUBASA

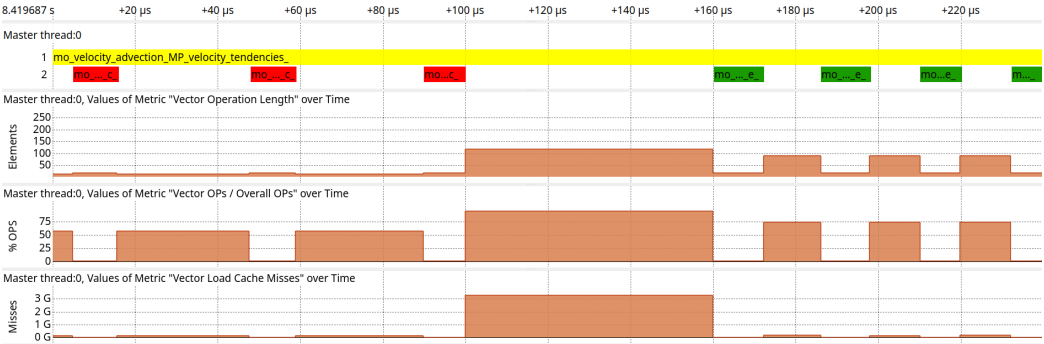
Score-P
Scalable performance measurement infrastructure for parallel codes

- **Invasive** measurement methods:
  - Automated function instrumentation, tool callback interfaces such as PMPI (MPI), OMPT (OpenMP), Cupti (CUDA), etc.
- **Requires recompilation** of the code with Score-P, but measurements can be adapted very well to the use case

lo2s

- **Non-invasive** measurement methods:
  - Instruction sampling, tracepoints, BPF, …
- Works with **unmodified applications**, but: Quality of measurements very dependent on quality of measurement interfaces

Example trace of the ICON weather model on NEC SX-Aurora TSUBASA with Score-P

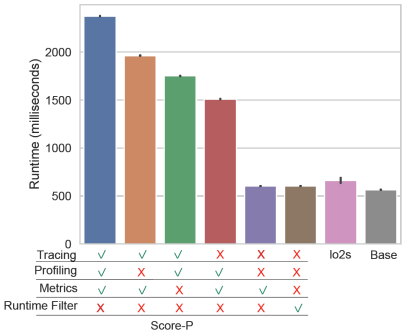


Graphic: C. v. Elm/ZIH

Details and Further Information

"Performance Tools for the NEC SX-Aurora TSUBASA" (C. v. Elm, R. Schöne, IPCE Companion '25)

Runtime Overhead of Different Tool Configurations for the ICON Weather Model



HPC at TU Dresden

- 6 clusters in total
- Barnard (2023, Eviden)
  - 720 nodes: 2x Intel Xeon Platinum 8470
  - 40 PB Lustre filesystem
- Capella (2024, Megware)
  - 156 nodes: 4x NVIDIA H100 + 2x AMD EPYC CPU 9334
  - 2 PB WEKA filesystem
  - #5 in the GREEN500; #51 in the TOP500 (Nov. 2024)

Capella GPU cluster (Foto: D. Hackenberg/ZIH)

CARA

- 2842 nodes (AMD Naples+Rome)
- Lustre *Bulk* (HDDs)
  - 16,5 PB / usage: 88%
  - `/storage/` and `/scratch`
- Lustre *Fast* (SSDs)
  - 500 TB / usage: 84%
  - `/home` and `/scratch_fast/`
- Users
  - On avg. 200 active users per month
  - 840 active users from 31 DLR institutes since start of operation
  - Production system with power users

Foto: DLR (CC BY-NC-ND 3.0)

Filesystem Capacity Usage over Time

General Discussion: Filesystems vs. Users

- Many different filesystems available
  - with different properties, strengths and features
- In the end, filesystems always fill up (faster than expected)
  - Not designed too small
  - Rather, because users don't clean up (and aren't required to)
- Data lifecycle management (cf. research data management)
  - Where are data generated and stored?
  - Which data are worth keeping? How long must they be retained?

Requirements for Filesystems

From operational point of view

1. Proper sizing at procurement
   - Adequately dimensioned to meet current and future needs
2. Controlled usage during operation
   - Filesystems must not be allowed to fill up uncontrollably during ongoing use

Possible Solutions for 2.

1. Fixed quota per user or project
   - Fair share (net capacity / #users)
   - Storage quota as part of the compute time application
   - Possibly with submit lock
2. Data management plan as part of the compute time application
   - Tools are lacking here

Practical Experience

- Users work at or way above their quota limit
- Increasing quota for VIPs
- Needs change over the course of the project
- Submit locks are not enforced (system utilization is prioritized)

Workspaces / Idea and Concept

Working Filesystemes

- Users have no default working directory (e.g., `/scratch/<login>`)
- Working filesystems can only be used via workspaces

Workspace

- A WS is a directory, with an associated expiration date, created on behalf of a user

Clean-up Mechanism

- *Expirer* mechanism controls lifetime of WSs
- Expired WSs will be moved; not accessible for users anymore
  - Deletion of data at end of grace period

Disclaimer

- Tool was developed by Holger Berger et al.
  - No merits for ZIH
- Experience report only, no “investment advice”
  - Improper configuration can lead to data loss
  - No liability can be accepted for data loss

HPC-Workspace

Implementierung

- github.com/holgerberger/hpc-workspace

Story

- Origination in 2004 at HLRS Stuttgart
- Solution to multliple challenges for the parallel filesystems of NEC SX-6 and SX-8
  - Gaining controll over lifetime of data with the goal to prevent very old data
  - Admin-controlled load balancing across multiple filesystems
  - Migration from one filesystem to replacement filesystem w/o user interaction

User View / Commands and Configuration

5+x basic commands

- **`ws_list`**: List active WSs and available filesystems
- **`ws_allocate`**: Create a WS
- **`ws_extend`**: Adjust lifetime of a WS
- **`ws_release`**: Mark a WS for deletion
- **`ws_restore`**: Restore an inactive WS

Example configuration

Filesystem	Time Limit	Extensions	Grace Period	Default Time
scratch	60 days	2 times	30 days	1 day
scratch_fast	30 days	2 times	30 days	1 day

Workspaces / Allocate

Command Syntax

```
$ ws_allocate [options] <workspace_name> <duration>
```

Valid characters for workspace names are only alphanumeric characters, -, ., and _.

Example

Allocate a workspace named *FancyExp* in */scratch* that will expire in 55 days:

```
$ ws_allocate -F scratch -n FancyExp -d 55
Info: creating workspace.
/scratch/ws/0/marie-FancyExp
remaining extensions  : 2
remaining time in days: 55
```

Workspaces / List

Oh, I don’t remember my workspaces.

```
$ ws_list
id: FancyExp
     workspace directory  : /scratch/ws/0/marie-FancyExp
     remaining time       : 54 days 23 hours
     creation time        : Mon Nov 30 10:04:57 2020
     expiration date      : Sun Jan 24 10:04:57 2021
     filesystem name      : scratch
     available extensions : 2
id: BestExp
    workspace directory  : /scratch/ws/0/marie-BestExp
    remaining time       : 14 days 23 hours
    ...
```

Workspaces / Automatic Release

- Workspace is released automatically at the end of its lifetime
- Workspace is moved to hidden directory
  - No data is deleted and data still occupies disk space (quota!)
    - Reminder: Data lifecycle
- Grace period starts (30 days)
  - Workspace can be restored using the command **`ws_restore`**
- After grace period
  - Workspace and its data are irretrievable deleted

Workspaces / Manual Release

- Workspaces can be released manually via

```
$ ws_release -n <workspace_name> -F <filesystem>
```

- Grace period for manually released workspaces is **only 1 day**
  - Workspace can be restored

Good Practice

- Use workspaces allocated from within batch jobs
- Delete data first, than release workspace

Workspaces / Various

- **`ws_allocate`** provides the functionality to send an email N days before expiration date, e.g.
  - `$ ws_allocate -n FancyExp -d 17 -r N -m marie@dlr.de`
- `ws_send_ical` sends an calender entry via email
- Cooperative WSs via **`ws_allocate`** `-g [-G <groupname>] foo 12`
- Multiple options to customize output of **`ws_list`**
- Manage links to all your workspaces using `ws_register`

Workspaces / Use Cases

- For campaign: Allocate and release once outside of batch scripts
- *Per-job storage:* Allocate and release within batch script

Workspaces / Admin View

- One YAML file per WS

```
$ cat /scratch/ws-db/marie-FancyExp
workspace: /scratch/ws/0/marie-FancyExp
expiration: 1729265074
extensions: 10
acctcode: hpcsupport
reminder: 7
mailaddress: marie@dlr.de
```

Workspaces / Admin View II

- Single configuration file: `/etc/ws.conf`

```
clustername: CARA                   # name to identify the system
smtphost: smtprelay.dlr.de
mail_from: root@cara.dlr.de
dbgid: 4                            # goup id, owner of workspace top-level directories
dbuid: 11                           # user id, owner of workspace top-level directories
admins: [root]                      # list of admins for ws_list
[...]                               # default life time settings
default: scratch
workspaces:
  scratch:                          # name of workspace location
    database: /scratch/ws-db        # DB directory
    deleted: .removed               # subdirectory for expired workspaces
    [...]                           # life time settings
    spaces: [/scratch/ws/]
  scratch_fast:
    database: /scratch_fast/ws-db
    deleted: .removed
    [...]
```

Workspaces / Admin View III

- Time configuration in `/etc/ws.conf`

```
[...]
duration: 60                    # max. duration in days
maxextensions: 2                # number of extensions
workspaces:
  scratch:
    [...]
    duration: 60
    keeptime: 30                # time in days to keep workspace after expiration
    maxextensions: 2
  scratch_fast:
    [...]
    duration: 30
    keeptime: 30
    maxextensions: 2
```

Additional Features for HPC Operation

Load-Balancing

```
workspaces:
  lustre:
    [...]
    spaces: [/lustre/ws/0, /lustre/ws/1]   # list of directories
    spaceselection: random                 # "random" (default), "uid" (uid%#spaces),
                                           # "gid" (gid%#spaces)
```

Migration of Filesystems

```
workspaces:
  lustre:
    [...]
    allocatable: no                  # do not allow new allocations in this workspace
    extendable: no                   # do not allow extensions in this workspace
    restorable: no                   # do not allow restores from this workspace
```

Lessons Learned and Experiences

- **Users**
  - High acceptance due to intuitive understanding
  - Scriptable + stable interface
- **`hpc-workspace` package**
  - (POSIX-compliant) filesystem agonistic
  - Open source and actively maintaned
  - Used by various HPC centers for quite a long time now
  - No alternatives™ on TU and DLR systems
- Interesting times:
  - **v2** is in development, c.f. https://github.com/holgerBerger/hpc-workspace-v2
  - New features; improved testing
  - A community would be desirable

Thank you for your attention.