natESM

PIK Leibniz Gemeinschaft

MAX PLANCK INSTITUTE OF GEOANTHROPOLOGY

PISM PARALLEL ICE SHEET MODEL

DKRZ Deutsches Klimarechenzentrum

JÜLICH Forschungszentrum | Jülich Supercomputing Centre

DKK Deutsches Klima Konsortium

Federal Ministry of Education and Research

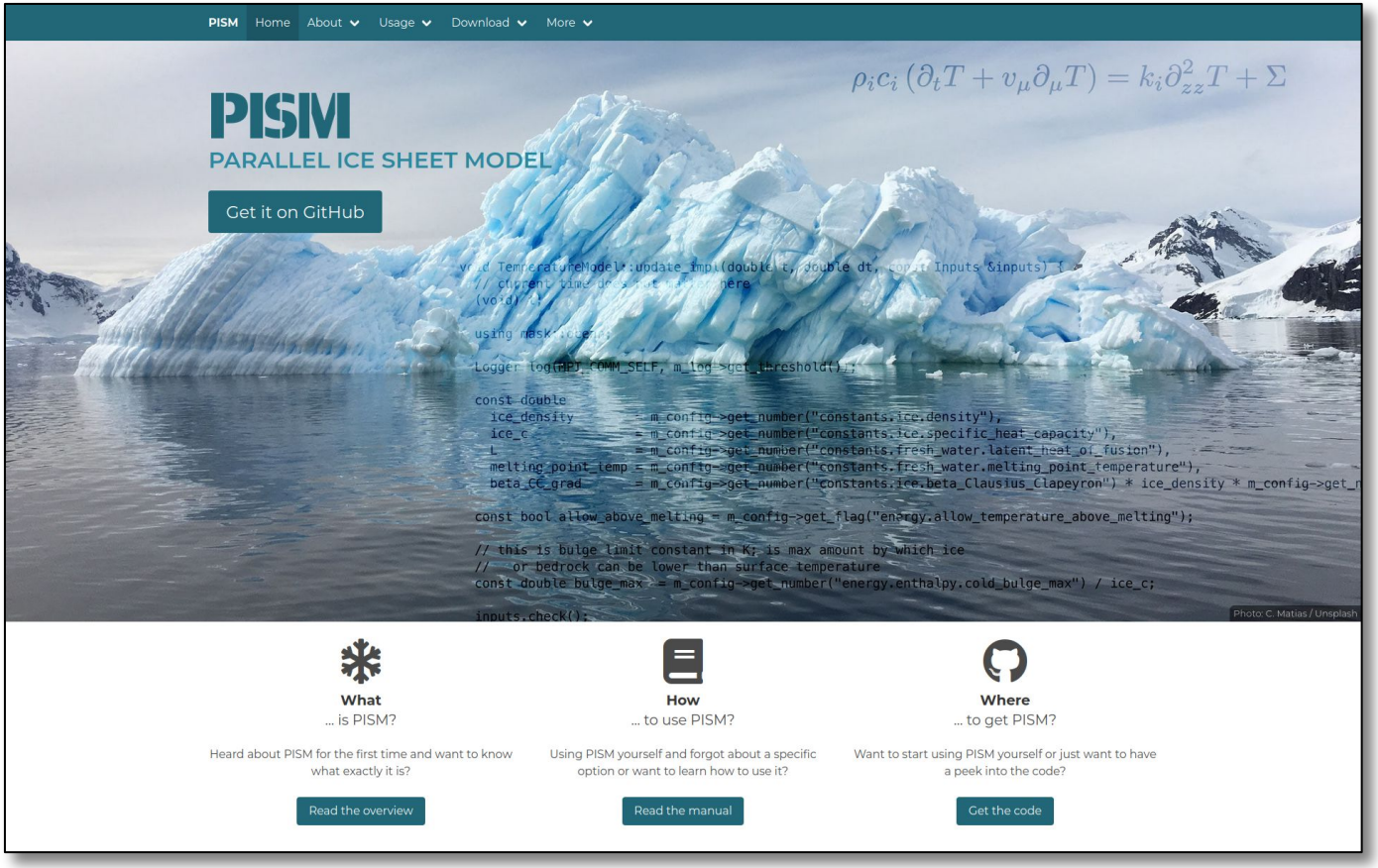
Sprint 17

Challenges and results experienced during the PISM sprint

April 2025 – October 2025

Torsten Albrecht (MPI-GEA/PIK), Wilton Jaciel Loch (DKRZ), Constantine Khrulev (UAF)

General information



Website
pism.io

Open-source
github.com/pism

Documentation
pism.io/docs

PISM
PARALLEL ICE SHEET MODEL

General information

Co-development at PIK since 2008 (Winkelmann et al., 2011) and at UAF, Alaska (Bueler & Brown, 2009) [1,2]

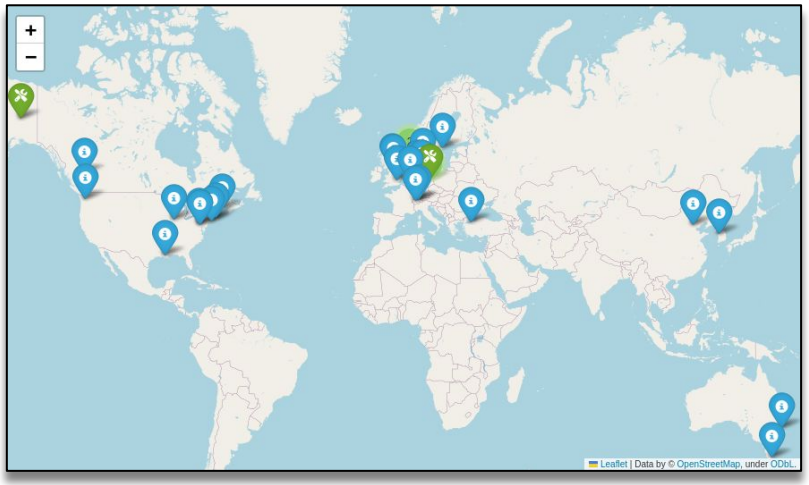
Many users and contributions worldwide (see map [3]) with more than 240 peer-reviewed publications [4]

Key research topics

- Interaction between Greenland and Antarctica via ocean, atmosphere and solid Earth
- Glacial cycle simulations and future sea-level projections
- Effect of extreme events on ice dynamics
- Tipping points and risk of tipping cascades

Technical details

- C++ code, MPI parallelization using PETSc toolkit
- interfaces to ocean, atmosphere and Earth components
- regular grid
- NetCDF based I/O

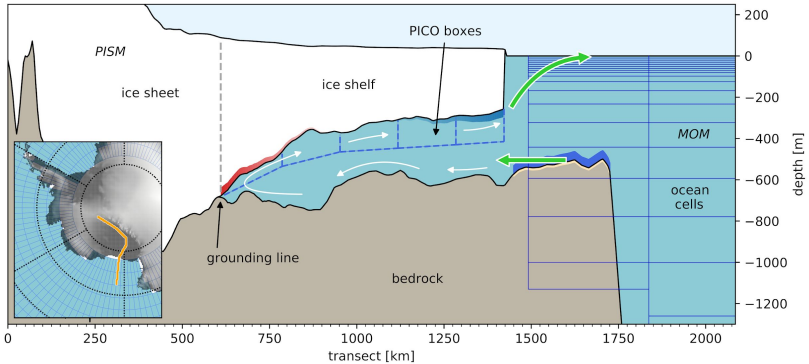


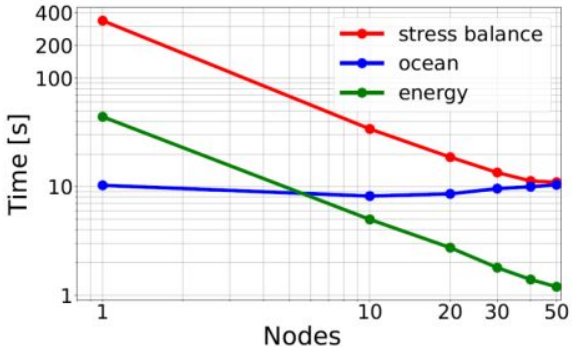
[1] https://www.pism.io/history
[2] https://www.pism.io/team
[3] https://www.pism.io/usersmap
[4] https://www.pism.io/publications

Sprint check

carried out by Enrico and Wilton (both DKRZ)

- scalability sprint preparing PISM for future high-resolution simulations (e.g. natESM integration)
    - Requirements for multi-million grid points
    - GPU compatibility requires in-depth analysis of PETSc performance





- Performance limitations identified for CPU:
    - limited scaling on CPU
    - I/O bottlenecks at high resolution
    - Flat strong scaling in PICO ocean component

PICO: Reese et al. (TC, 2018), Kreuzer et al., (GMD, 2021)

Challenges - Asynchronous Output Server

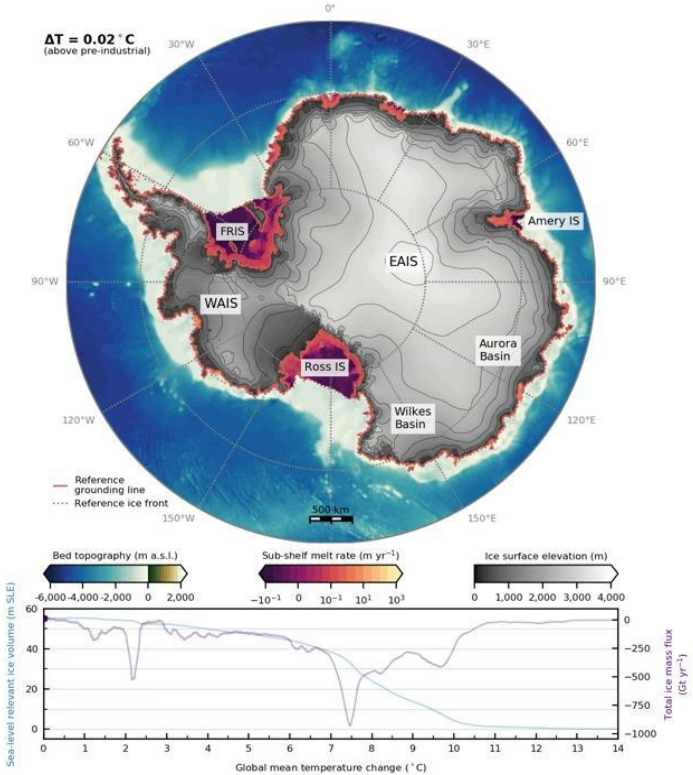
Problem: Synchronous I/O Bottleneck

Simulation pauses during output of different file classes:

- timeseries (1D, high time resolution)
- extra (2D or 3D diagnostics)
- snapshots/backups (PISM restartable 3D)
  - extended grid for bed deformation

Output time grows linearly with simulation length

At high resolution → severe performance penalty, as I/O dominates runtime in long simulations.



Tipping in Antarctica: Garbe et al. (Nature, 2020), Winkelmann et al., (NCC, 2026)

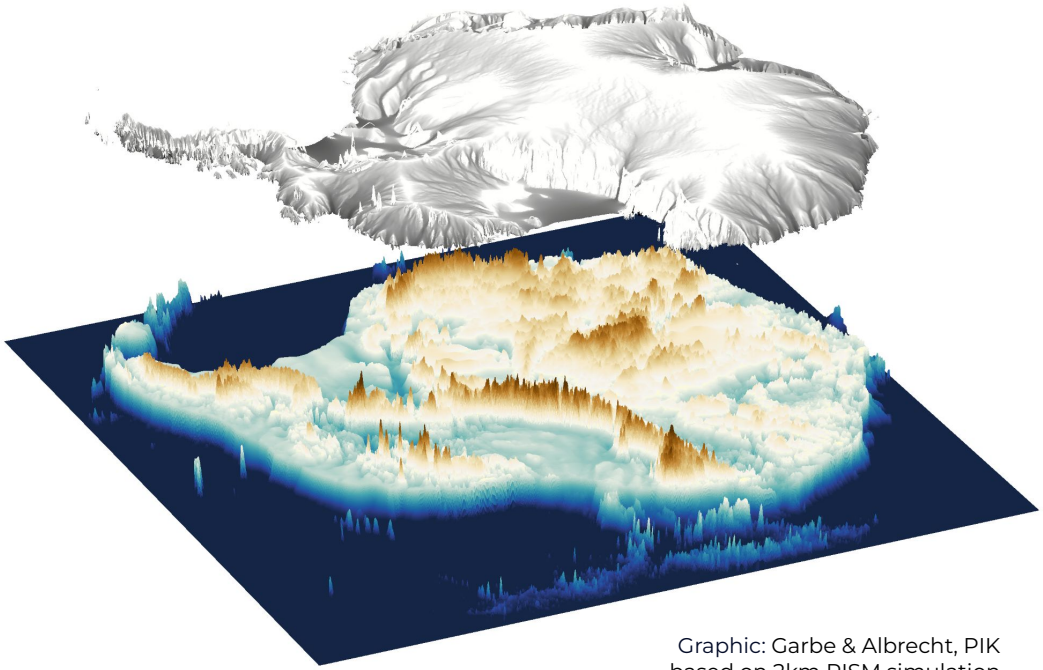
Challenges - Asynchronous Output Server

Goals:

- Decouple simulation from file writing
- Preserve existing output structure
- Allow runtime switch (sync vs. async)
- Keep flexibility for future coupling

Tools:

- Evaluated: ADIOS2, YAXT and YAC
- Chosen: YAC
  - Already integrated in PISM
  - Multi-language support
  - Metadata & time management
  - Flexible design
  - Yet, YAC not designed as output library



Graphic: Garbe & Albrecht, PIK
based on 2km PISM simulation

Major Technical Challenges - Asynchronous Output Server

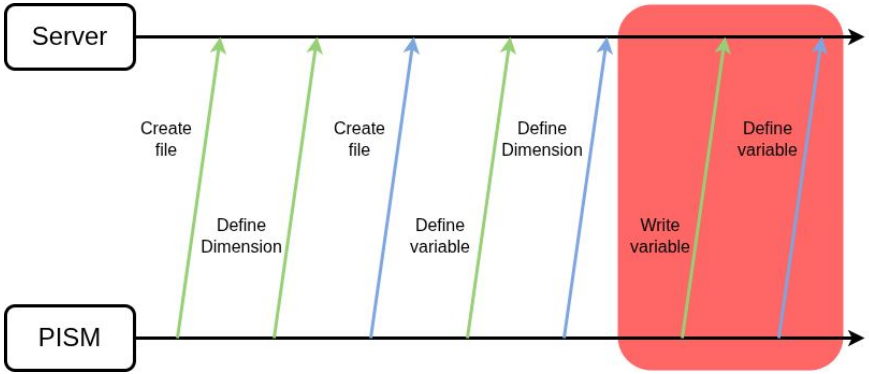
Grid information availability:

- Grid only exists inside PISM at runtime
- Not guaranteed in external files
- Needed to transfer grid metadata via MPI intercommunicator

Interleaved Output Operations (core architectural conflict):

- PISM:
  - Arbitrary order of file operations
  - Multiple files handled interleaved
- YAC:
  - Strict phase-based design
  - Definitions phase
  - Data exchange phase

➡ Deadlock risk



One possible order of file operations performed on PISM. Different arrow colors represent different file classes and the red region represents the conflict for the output server.

Results - Solution for Asynchronous Output Server

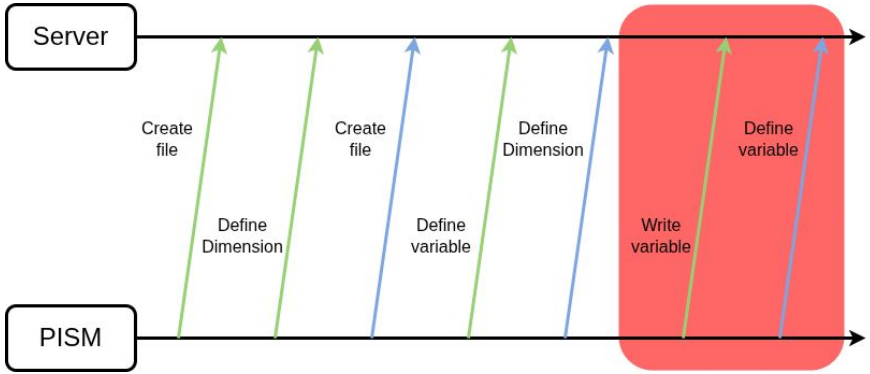
Key developments:

- ✔ MPI intercommunicator between PISM & server
- ✔ Action-based client–server communication
- ✔ Redesign of PISM output interface
- ✔ Pre-definition of variables to avoid YAC deadlocks

Result:

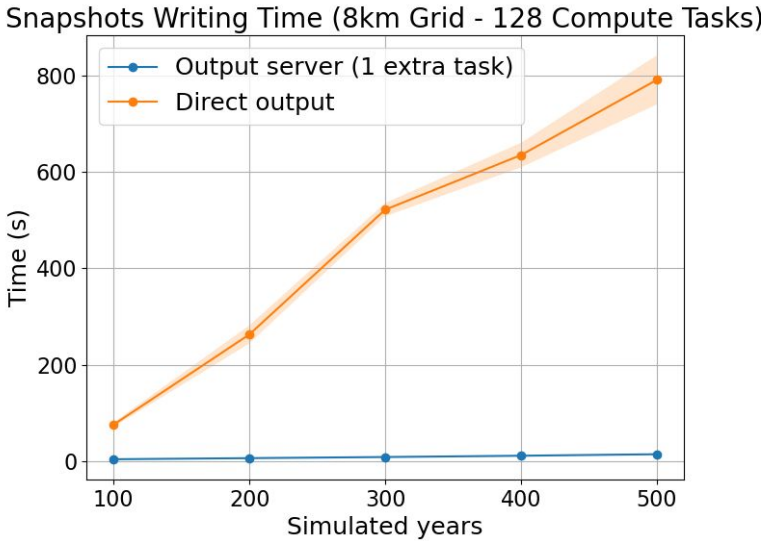
- Flexible action-driven server
- Snapshot support fully working

➡ Infrastructure now reusable beyond this sprint.



One possible order of file operations performed on PISM. Different arrow colors represent different file classes and the red region represents the conflict for the output server.

Results - Performance for Asynchronous Output Server



Total snapshot writing time for the original synchronous output and the asynchronous output server with different simulation lengths.

Async Output Performance

Experiment - 8km Antarctica (760 x 760):

- Snapshots every 5 simulated years
- Single node + 1 server task
- Averaged over 5 runs

Result:

- ~98% reduction in snapshot writing time
- Benefit increases with simulation length

Challenges - PICO ocean module performance

Technical Challenge: Original Eikonal solver:

- Approximate discrete algorithm
- Many unnecessary loops & checks
- Inefficient domain traversal

Implemented:

- BFS-based algorithm
- Efficient domain updates

Results:

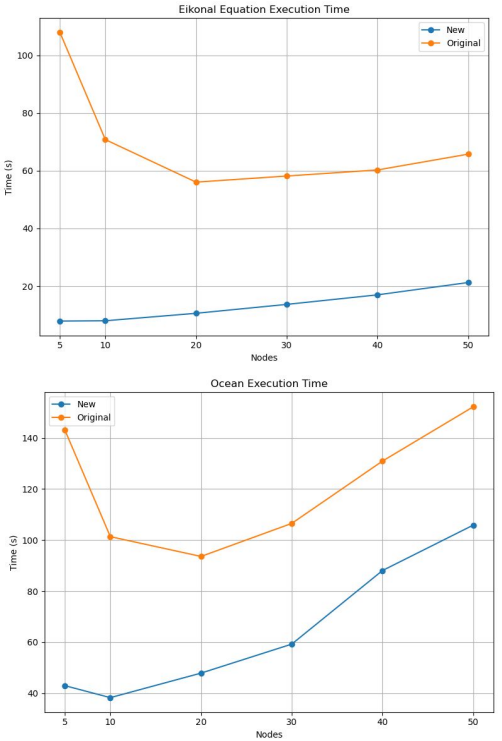
Eikonal solver:

- 80–98% runtime reduction

Whole ocean component:

- 62–69% runtime reduction

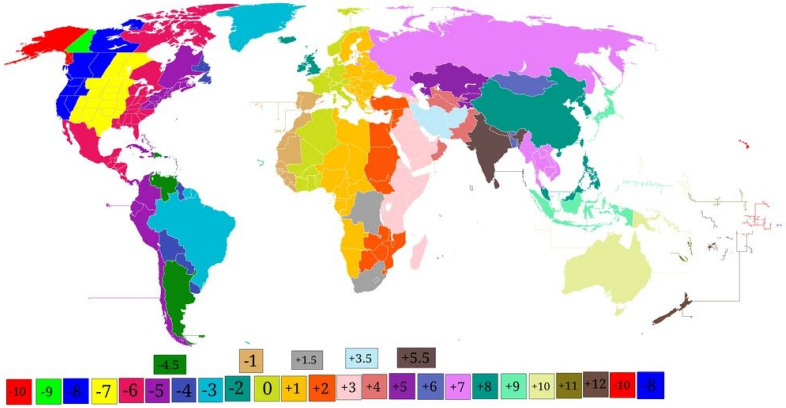
➡ Significant improvement with moderate implementation effort.



Eikonal equation solution time (top) and total PICO execution time (bottom) for original and newly implemented algorithms.

Non-technical Challenges

- Time-zone differences (Germany–Alaska)
- Dependency on external feedback, but limited availability of maintaining institution



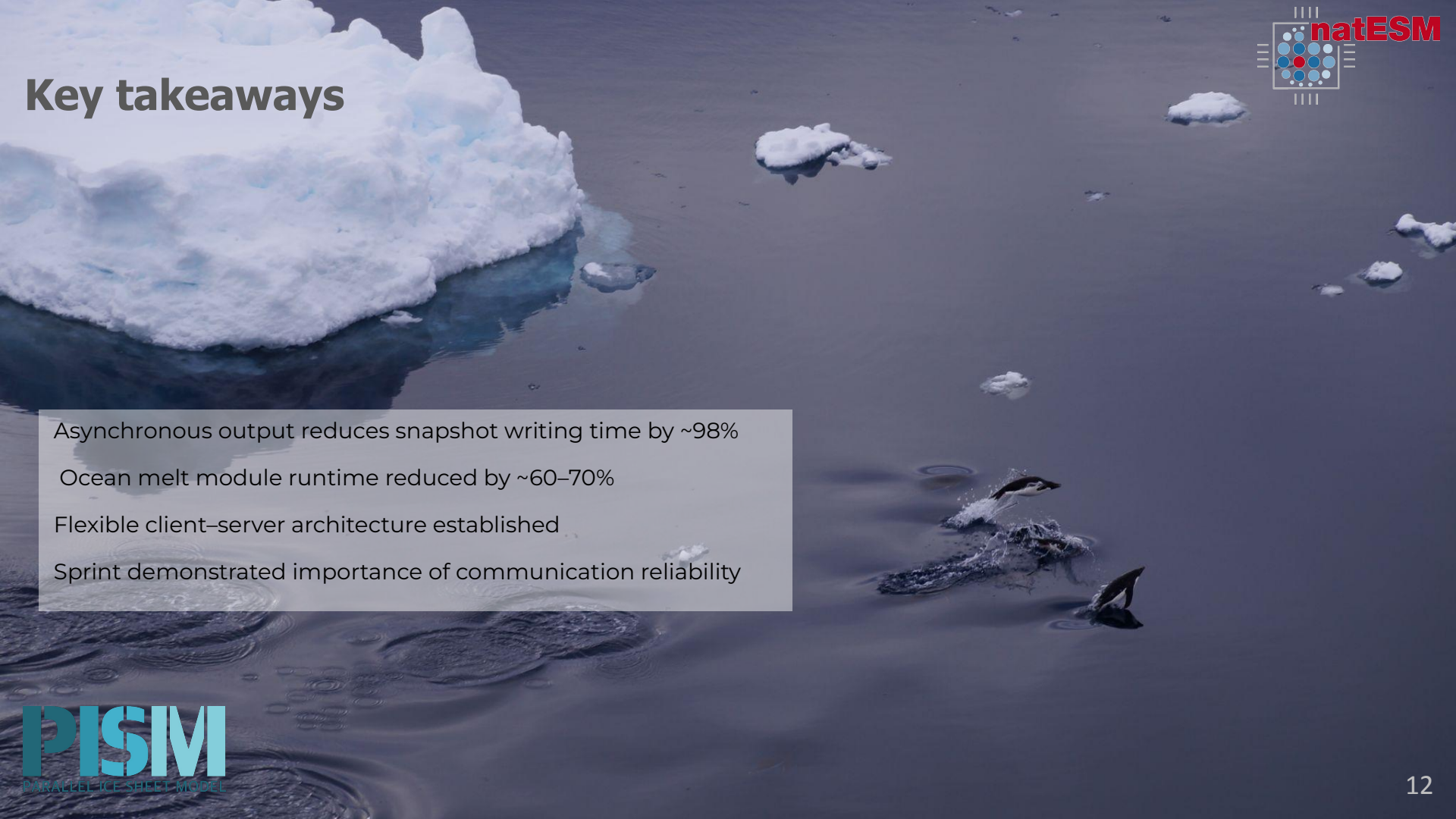
https://calendars.fandom.com/wiki/Time_zone?file=Timezones_optimized.png

Mitigation:

- definition of multiple, partially independent goals helped to reduce the impact of such interruptions
- RSE can continue work on alternative tasks whenever progress on the primary objective

➡ Important lesson for future collaborations

Key takeaways



- Asynchronous output reduces snapshot writing time by ~98%
- Ocean melt module runtime reduced by ~60–70%
- Flexible client–server architecture established
- Sprint demonstrated importance of communication reliability