

Beyond CI/CD: Continuous Performance-Driven Principles for Future HPC Systems and Applications

Yoshifumi Nakamura

RIKEN R-CCS

Operations and Computer Technologies Division, Software Development Technology Unit

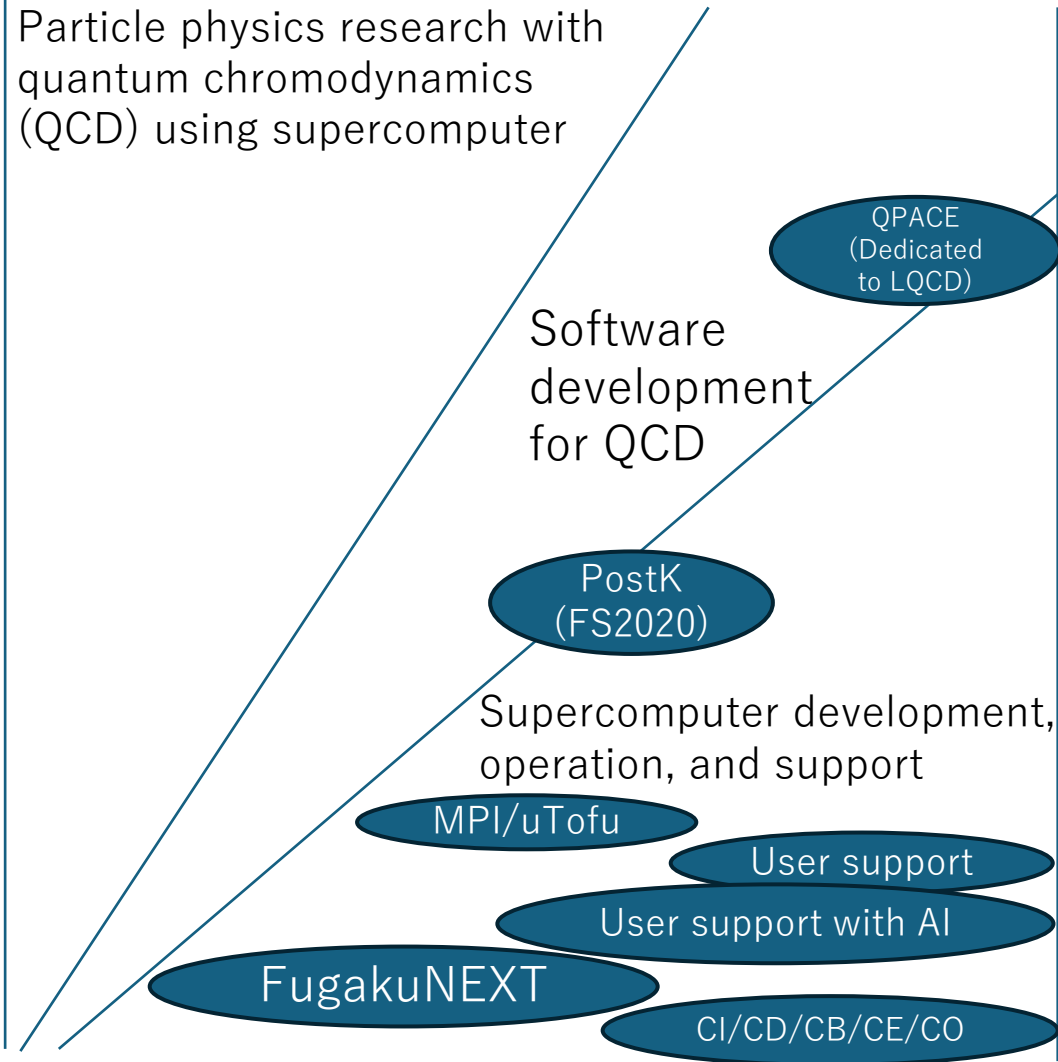
FugakuNEXT project, Communication library WG Leader / Benchmark WG Sub-leader

The 2nd “FugakuNEXT” Application Seminar / 第2回「富岳NEXT」アプリケーションセミナー

July 17, 2025

About my work history

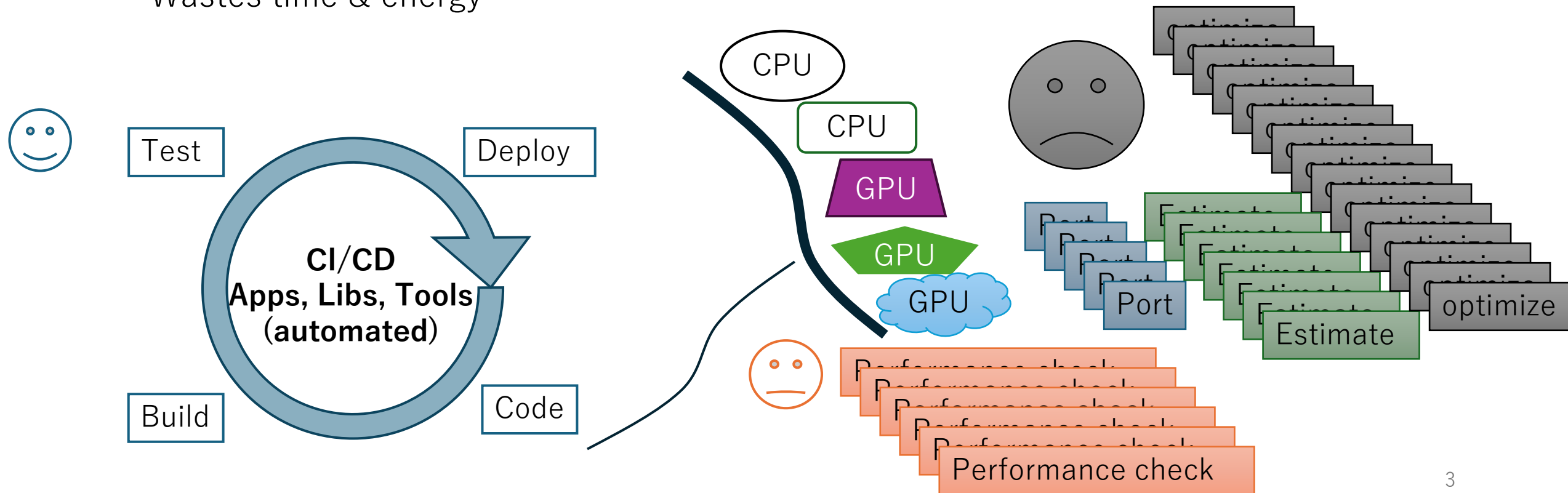
Particle physics research with quantum chromodynamics (QCD) using supercomputer



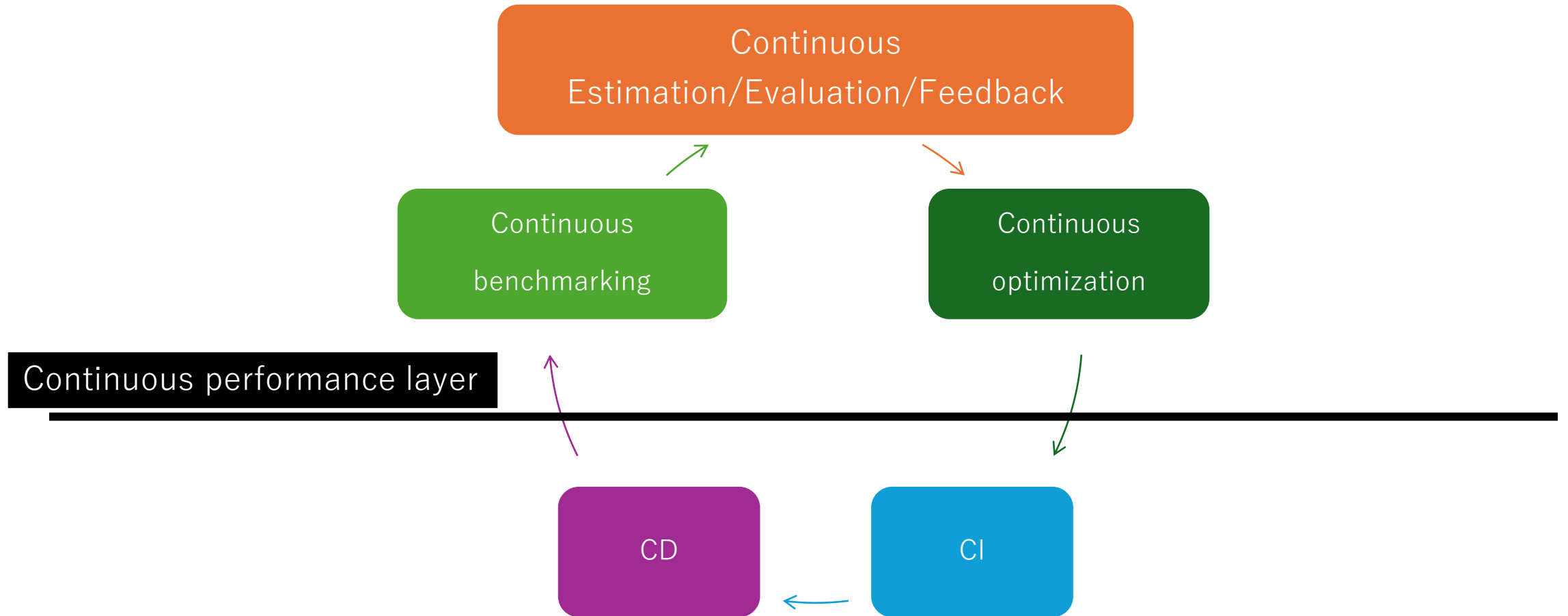
- Deutsches Elektronen-Synchrotron DESY
 - 2005 – 2008
- University of Regensburg
 - 2008 – 2010
- University of Tsukuba
 - 2010 - 2011
- RIKEN
 - 2011 –

Why CI/CD Alone Is Not Enough for HPC

- Continuous Integration/ Continuous Delivery/Deployment (CI/CD) is common
- Performance checks are late, manual
- Wastes time & energy



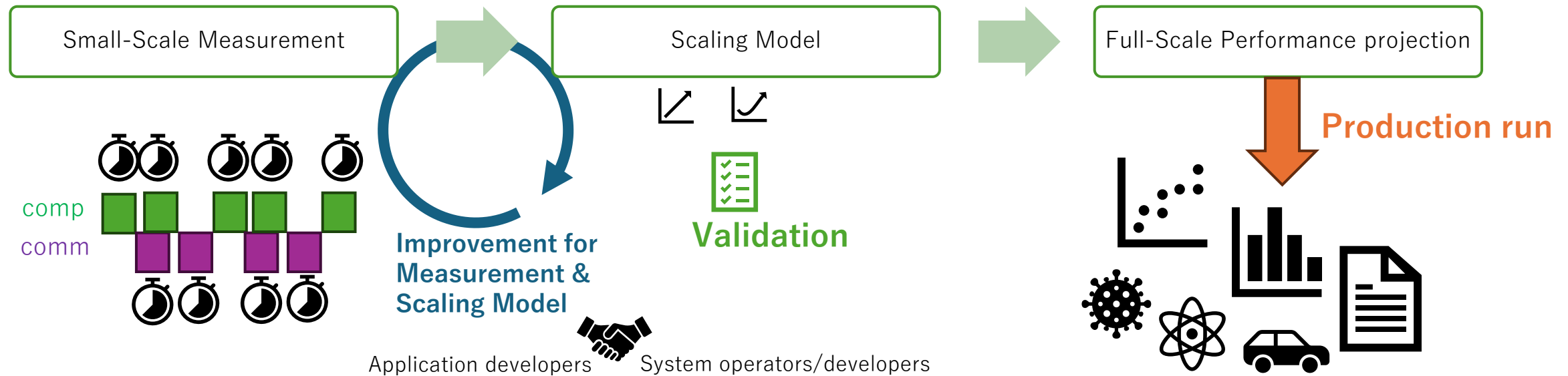
Continuous Performance-Driven Principles



Goal: Performance awareness and improvement become routine, not special isolated tasks.

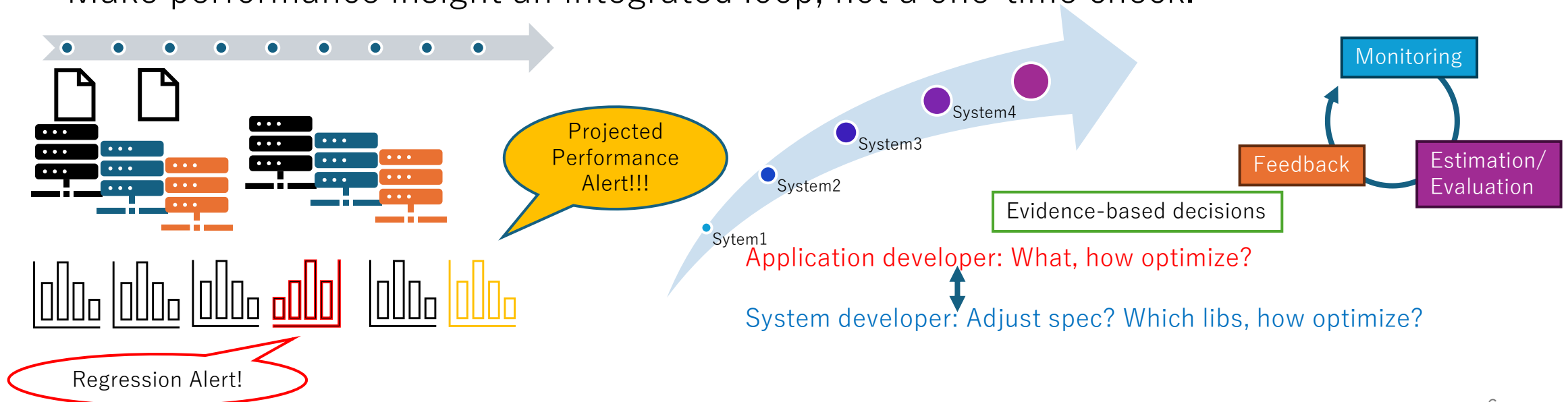
Principle 1: Meaningful & Validated Performance Measurement

- Measure performance in sections matching optimization scopes.
- Use realistic or properly scaled input conditions.
- Ensure performance measurement results are credible for real production.
- Support trustworthy projections for future HPC systems.



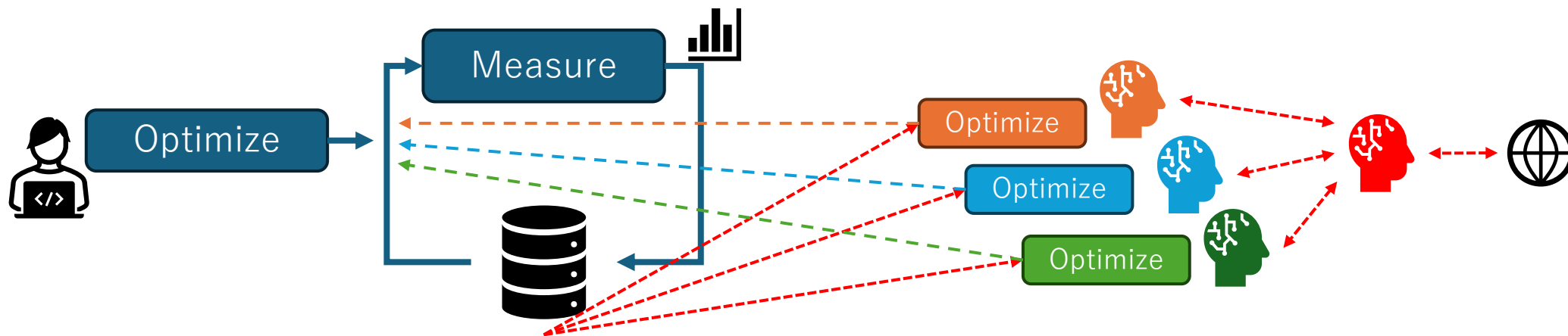
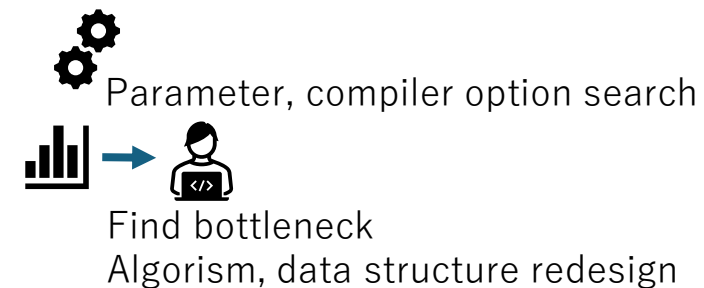
Principle 2: Continuous Performance Monitoring & Feedback

- Continuously collect and observe performance data during real workloads.
- Evaluate results to detect deviations and unexpected trends.
- Estimate future performance under realistic conditions.
- Provide clear, actionable feedback to both application and system teams.
- Make performance insight an integrated loop, not a one-time check.



Principle 3: Continuous Optimization

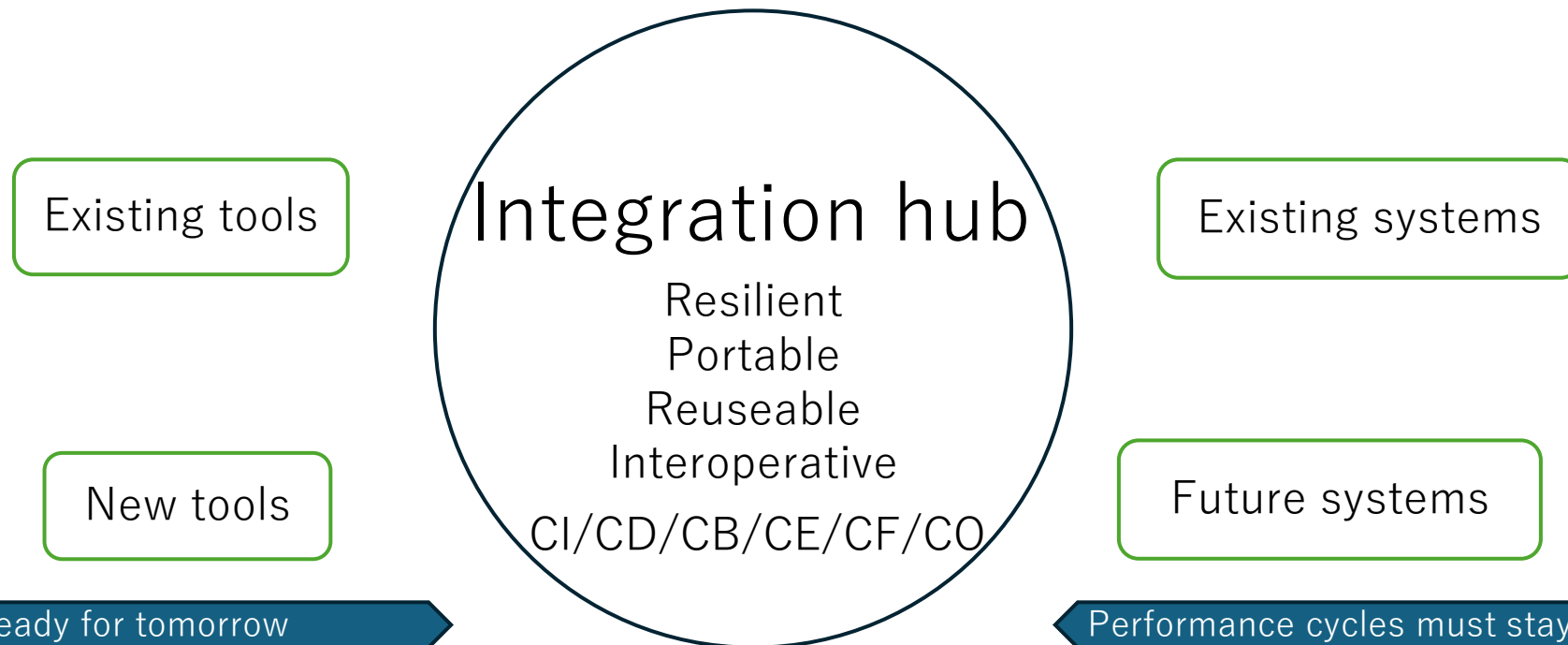
- Optimization must shift from one-time to continuous.
- Start today: measure more, test variations, record results.
- Automate what you can, guide developers with clear data.
- Build a performance history for AI auto-tuning.



Today's groundwork unlocks tomorrow's AI-driven performance

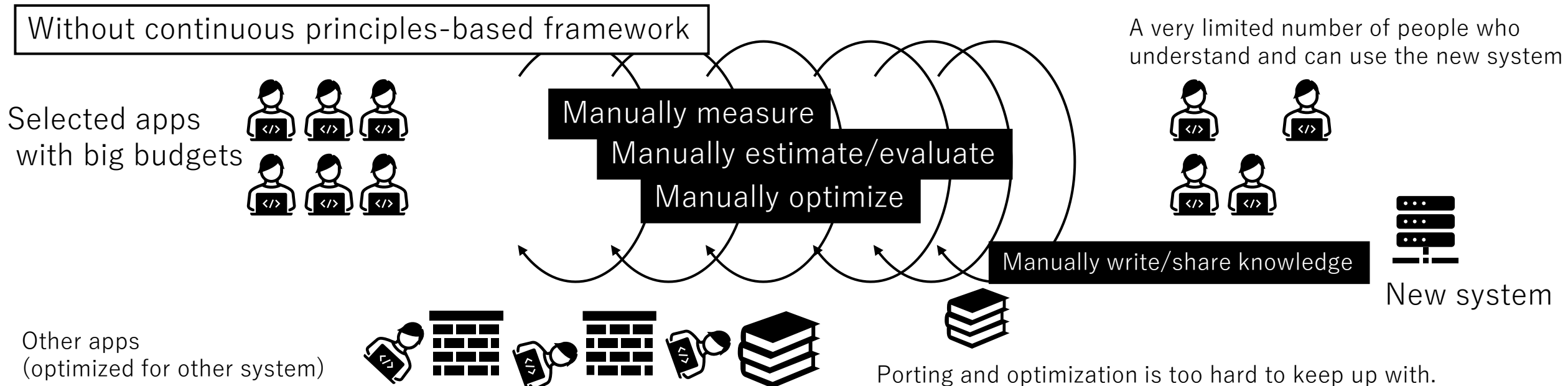
Principle 4: Open, Flexible, and Sustainable Integration

- Support diversity: HPC systems, tools, teams
- Design for change: Stay adaptable, avoid lock-in
- Enable reuse: Open formats & clear interfaces



Why This Matters Now for Supercomputer Development

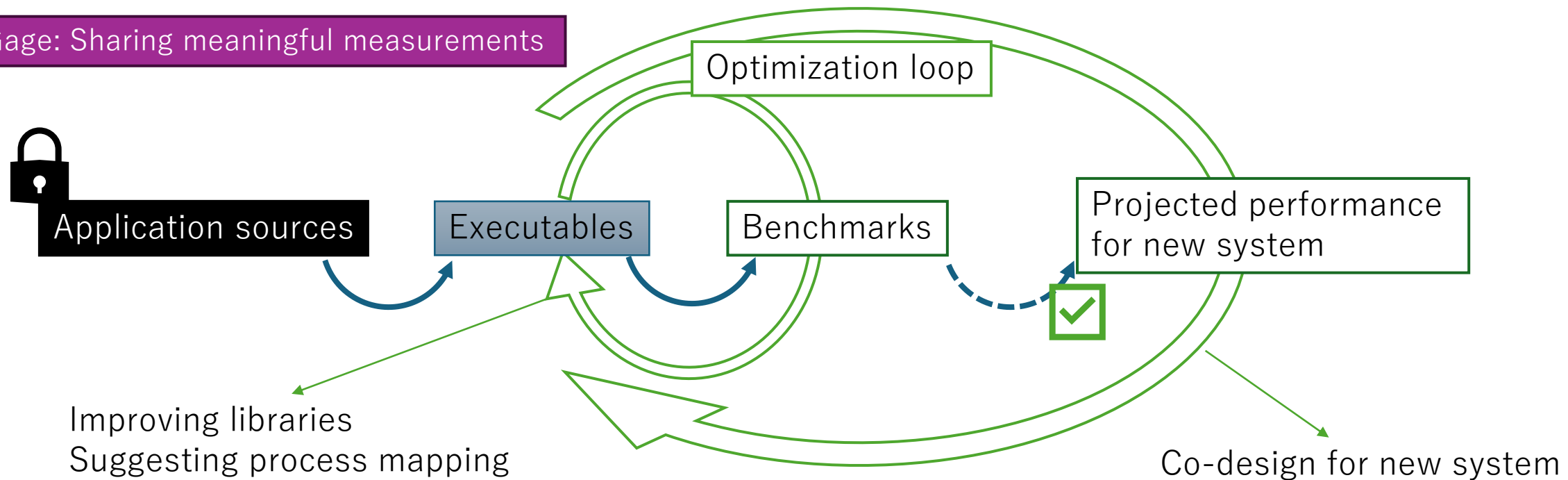
- Today's supercomputers are more complex and use more mixed hardware than ever.
- Hardware and software are still often designed separately. It causes problems.
- We need performance checks and feedback from the beginning.
- This helps design hardware that really fits real apps and workloads.



Collaborate and Protect: Open Performance, Private Code

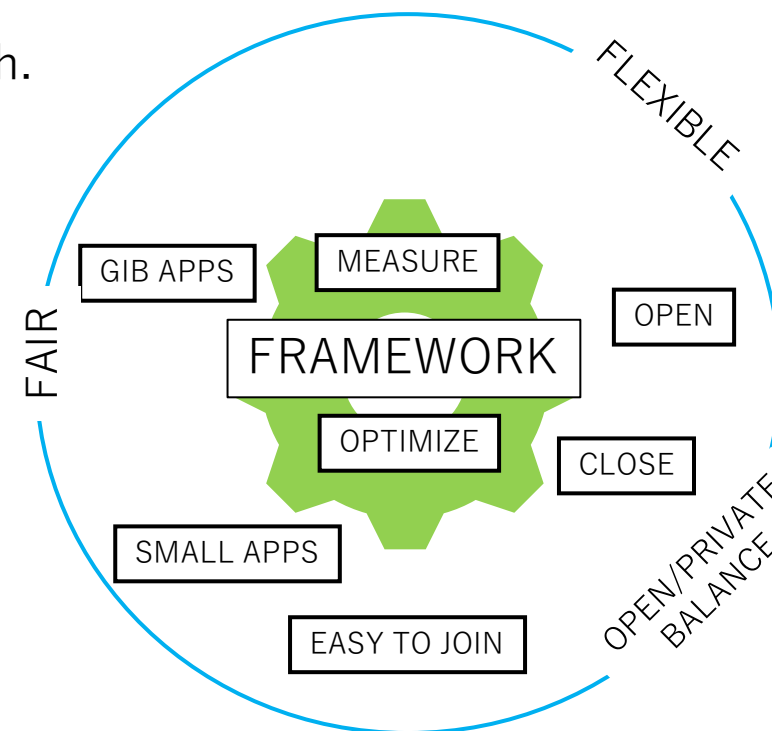
- Full source openness isn't always possible, but sharing key performance data is.
- By sharing benchmarks, both open and closed projects stay in the optimization loop.

Minimum Gage: Sharing meaningful measurements



Making Performance Work for Everyone

- Many developers are not performance experts.
- The framework should measure and share useful data easily.
- Sharing all code is not needed, sharing performance just enough.
- It should work with little extra effort or skills.
- Openness and privacy must balance.
- Make co-design simple and fair for big and small projects.
- Joining late should still help and benefit all.



Conclusion

- CI/CD alone is not enough, go beyond.
- Add benchmarking, estimation, evaluation, feedback, and optimization.
- 4 principles guide sustainable progress.
- Big or small, open or closed, all apps can grow with HPC.
- We should review these principles as technology and needs change.

Thank you for your attention