Karlsruhe Institute of Technology

Ginkgo

How do we make sure our code does not get worse?

Towards Continuous Benchmarking (CB)

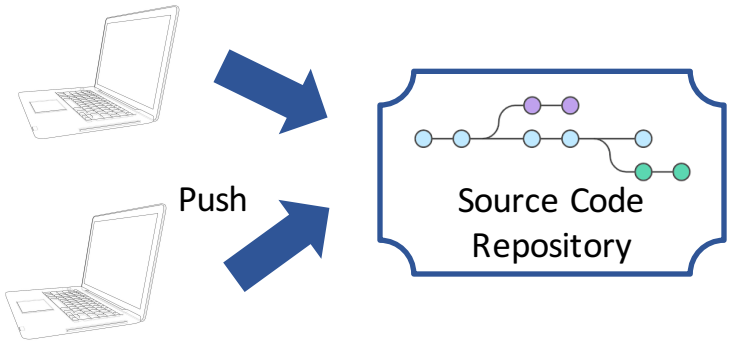
GridKa School 2019 – The Art of Data
August 26th-30th, 2019 | Karlsruhe, Germany

Hartwig Anzt, Terry Cojean, Goran Flegar, Thomas Grützmacher, Pratik Nayak, Mike Tsai

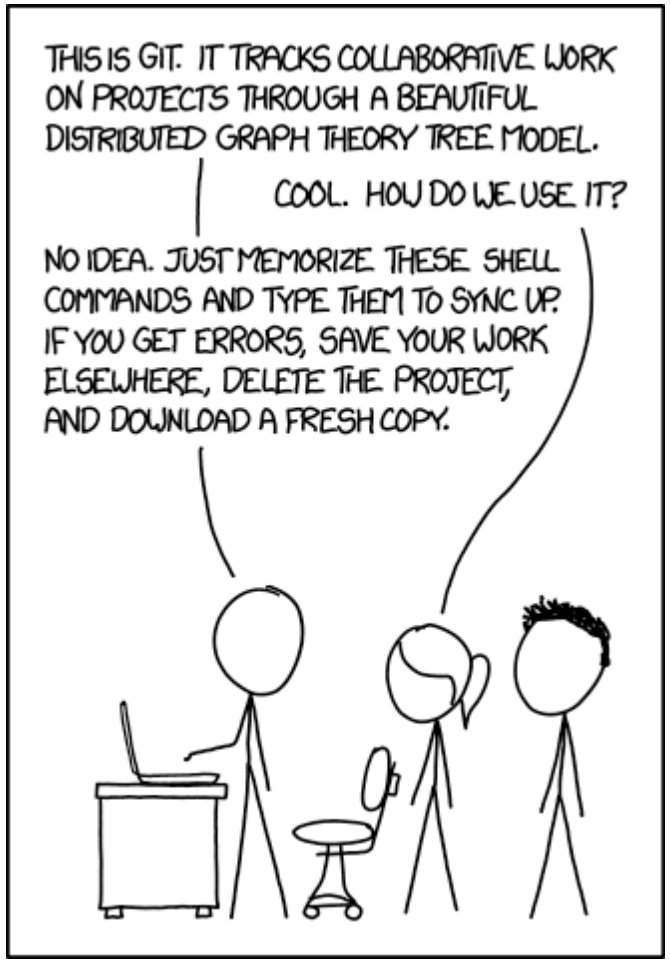
A Healthy Software Development Cycle

Developer

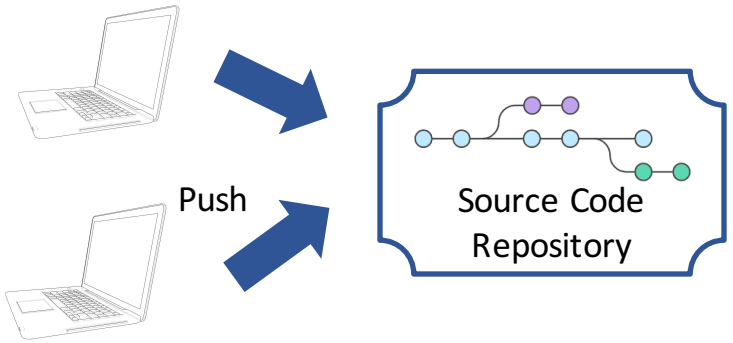
A Healthy Software Development Cycle



- Version control system for tracking changes and coordinating collaborative development.



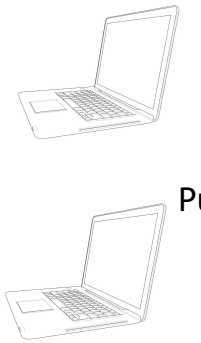
A Healthy Software Development Cycle



- Version control system for tracking changes and coordinating collaborative development.

1. *Is my software working (compiling)?*
2. *Is my software working (does the correct things)?*
3. *Is my software working fast? (the High in HPC)?*

A Healthy Software Development Cycle



Developer

Push

Source Code Repository

CMake Cross-platform Make

CI Build

Continuous Integration (CI)

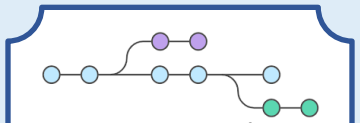
- Version control system for tracking changes and coordinating collaborative development.
- CI Build system continuously checks the compilation success on different architectures.

1. *Is my software working (compiling)?* ✓
2. *Is my software working (does the correct things)?*
3. *Is my software working fast? (the High in HPC)?*

A Healthy Software Development Cycle

Push

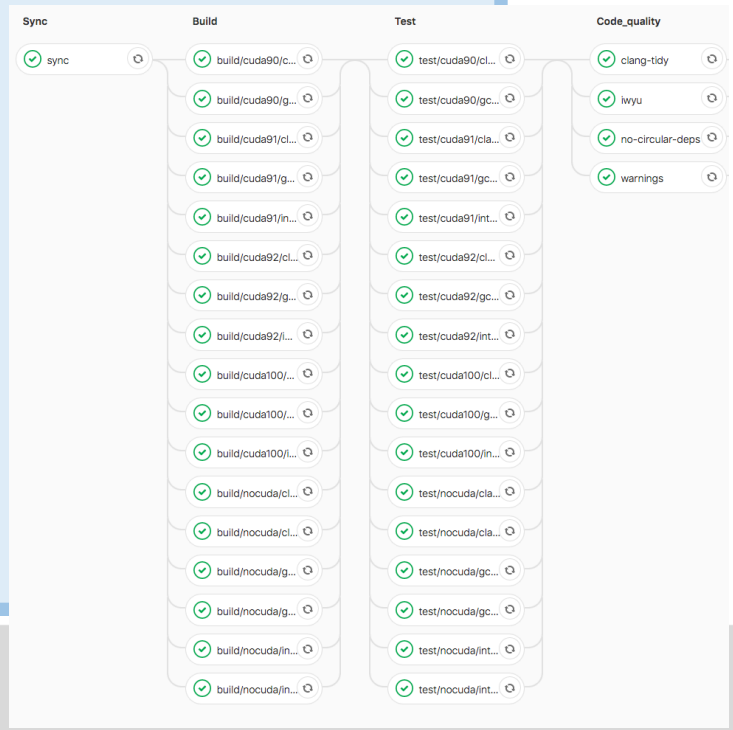
Developer



Continuous Integration (CI)

- Version control system for tracking changes and coordinating collaborative development.
- CI Build system continuously checks the compilation success on different architectures.

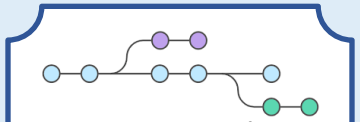
1. *Is my software working (compiling)?* ✓
2. *Is my software working (does the correct things)?*
3. *Is my software working fast? (the High in HPC)?*



A Healthy Software Development Cycle

Push

Developer



Continuous Integration (CI)

- Version control system for tracking changes and coordinating collaborative development.
- CI Build system continuously checks the compilation success on different architectures.
- Unit tests ensure functionality and validity of all building blocks.

Continuous 10 builds [view timeline]

		Configure		Build		Test				
Site	Build Name	Error	Warn	Error	Warn	Not Run	Fail	Pass	Start Time	Labels
Linux-FineCI	COVERAGE	0	9	0	13	0	0	103	1 hour ago	(none)
Linux-FineCI	COVERAGE	0	9	0	13	0	0	103	1 hour ago	(none)
Linux-FineCI	Valgrind	0	9	0	13				2 hours ago	(none)
Linux-FineCI	ASAN	0	5	0	1				2 hours ago	(none)
Linux-FineCI	TSAN	0	5	0	1				2 hours ago	(none)
Linux-FineCI	COVERAGE	0	9	0	13	0	0	103	2 hours ago	(none)
Linux-FineCI	ASAN	0	5	0	1				5 hours ago	(none)
Linux-FineCI	Valgrind	0	9	0	13				5 hours ago	(none)
Linux-FineCI	TSAN	0	5	0	1				5 hours ago	(none)
Linux-FineCI	COVERAGE	0	9	0	13	0	0	103	6 hours ago	(none)

Coverage

Site	Build Name	Percentage	LOC Tested	LOC Untested	Date	Labels
Linux-FineCI	COVERAGE	95.97%	6737	283	6 hours ago	(none)
Linux-FineCI	COVERAGE	95.97%	6737	283	2 hours ago	(none)
Linux-FineCI	COVERAGE	95.97%	6737	283	1 hour ago	(none)
Linux-FineCI	COVERAGE	95.97%	6737	283	1 hour ago	(none)

Dynamic Analysis

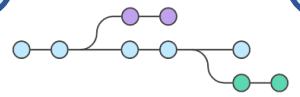
Site	Build Name	Checker	Defect Count	Date
Linux-FineCI	TSAN	ThreadSanitizer	3178	5 hours ago
Linux-FineCI	Valgrind	Valgrind	80	5 hours ago
Linux-FineCI	ASAN	AddressSanitizer	0	5 hours ago
Linux-FineCI	TSAN	ThreadSanitizer	3184	2 hours ago
Linux-FineCI	ASAN	AddressSanitizer	0	2 hours ago

1. *Is my software working (compiling)?* ✓
2. *Is my software working (does the correct things)?* ✓
3. *Is my software working fast? (the High in HPC)?*

A Healthy Software Development Cycle

Push

Developer



Source Code Repository

CMake Cross-platform Make

CI Build

googletest Google C++ Testing Framework

CI Test

Continuous Integration (CI)

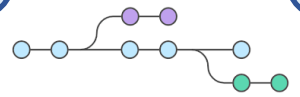
- Version control system for tracking changes and coordinating collaborative development.
- CI Build system continuously checks the compilation success on different architectures.
- Unit tests ensure functionality and validity of all building blocks.
- *In HPC software development, we want to make sure new contributions do not degrade performance!*

1. *Is my software working (compiling)?* ✓
2. *Is my software working (does the correct things)?* ✓
3. *Is my software working fast? (the High in HPC)?*

A Healthy Software Development Cycle

Push

Developer



Source Code Repository

CMake Cross-platform Make

CI Build

googletest Google C++ Testing Framework

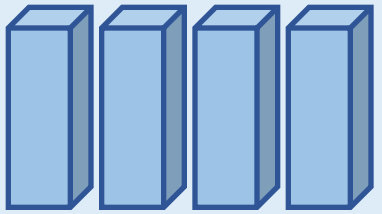
CI Test

Continuous Integration (CI)

- Version control system for tracking changes and coordinating collaborative development.
- CI Build system continuously checks the compilation success on different architectures.
- Unit tests ensure functionality and validity of all building blocks.
- *In HPC software development, we want to make sure new contributions do not degrade performance!*
- Idea: Automated benchmark runs every time new code is added.

CI Benchmark Tests

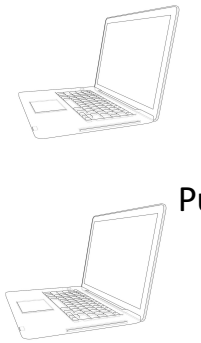
Schedule in Batch System



HPC System

1. *Is my software working (compiling)?* ✓
2. *Is my software working (does the correct things)?* ✓
3. *Is my software working fast? (the High in HPC)?*

A Healthy Software Development Cycle



Push

Developer

Source Code Repository

CMake Cross-platform Make

CI Build

googletest Google C++ Testing Framework

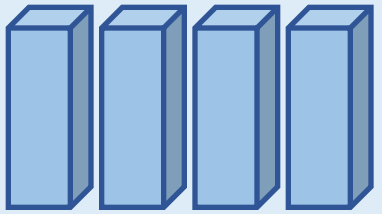
CI Test

Continuous Integration (CI)

- Version control system for tracking changes and coordinating collaborative development.
- CI Build system continuously checks the compilation success on different architectures.
- Unit tests ensure functionality and validity of all building blocks.
- *In HPC software development, we want to make sure new contributions do not degrade performance!*
- Idea: Automated benchmark runs every time new code is added.

CI Benchmark Tests

Schedule in Batch System

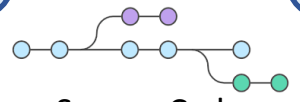


HPC System

A Healthy Software Development Cycle

Push

Developer



Source Code Repository

CMake Cross-platform Make

CI Build

googletest Google C++ Testing Framework

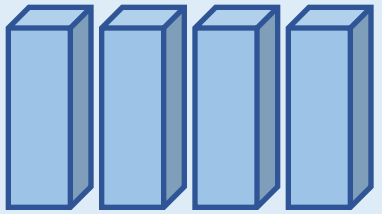
CI Test

Continuous Integration (CI)

- Version control system for tracking changes and coordinating collaborative development.
- CI Build system continuously checks the compilation success on different architectures.
- Unit tests ensure functionality and validity of all building blocks.
- *In HPC software development, we want to make sure new contributions do not degrade performance!*
- Idea: Automated benchmark runs every time new code is added.

CI Benchmark Tests

Schedule in Batch System

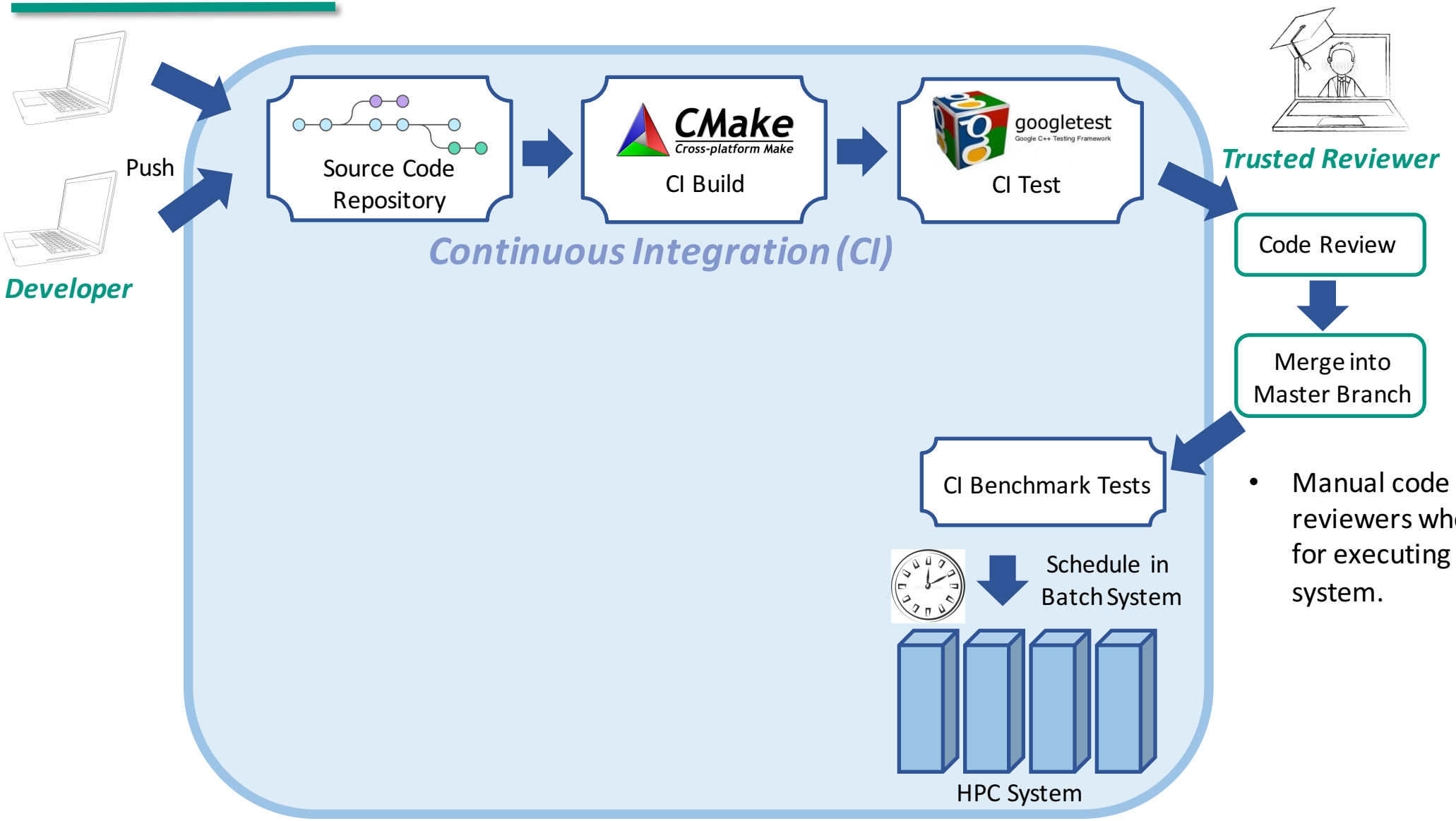


HPC System

Security Thrust!

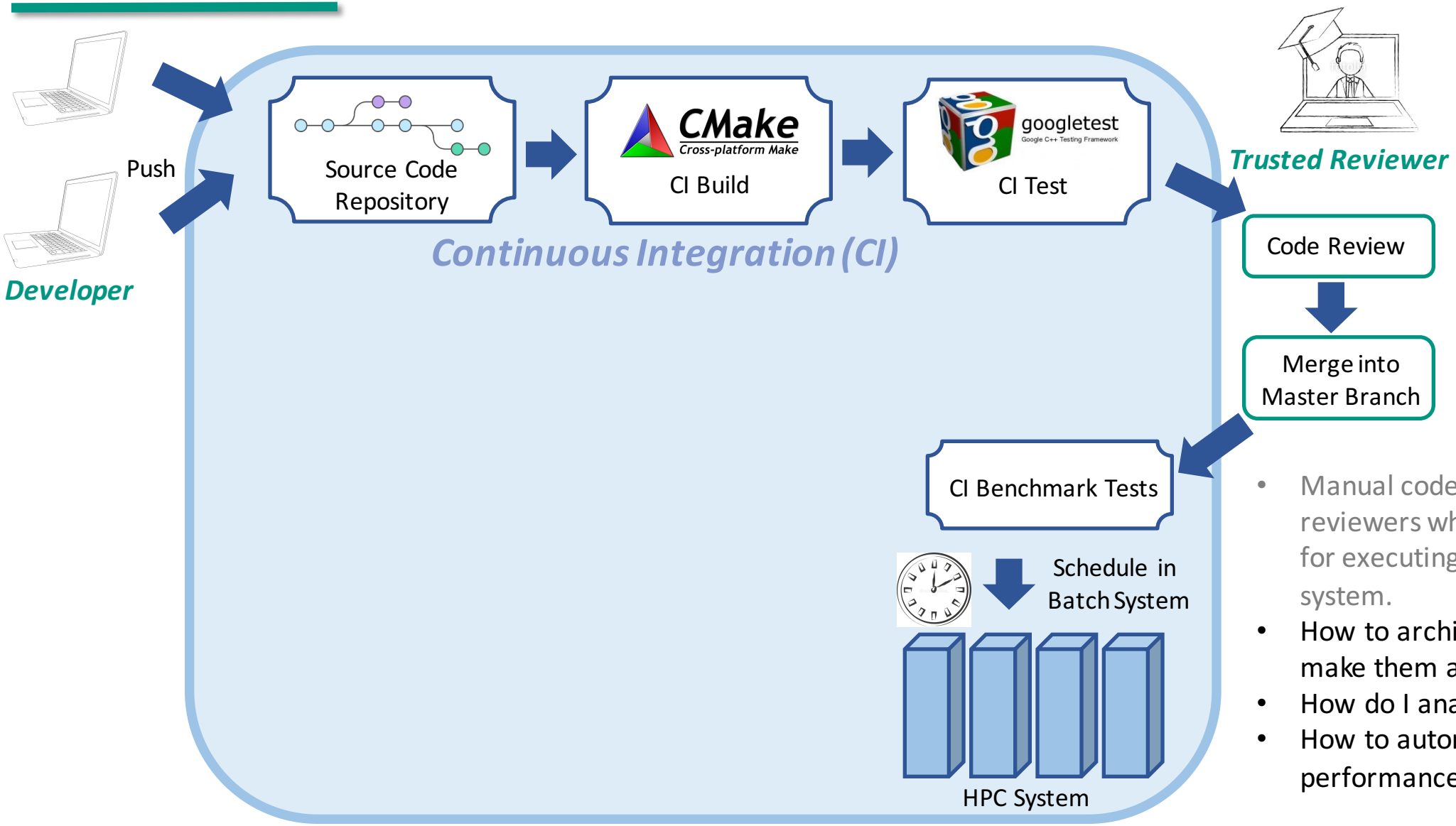
- External code is run on HPC system
- Possible point of attack

A Healthy Software Development Cycle



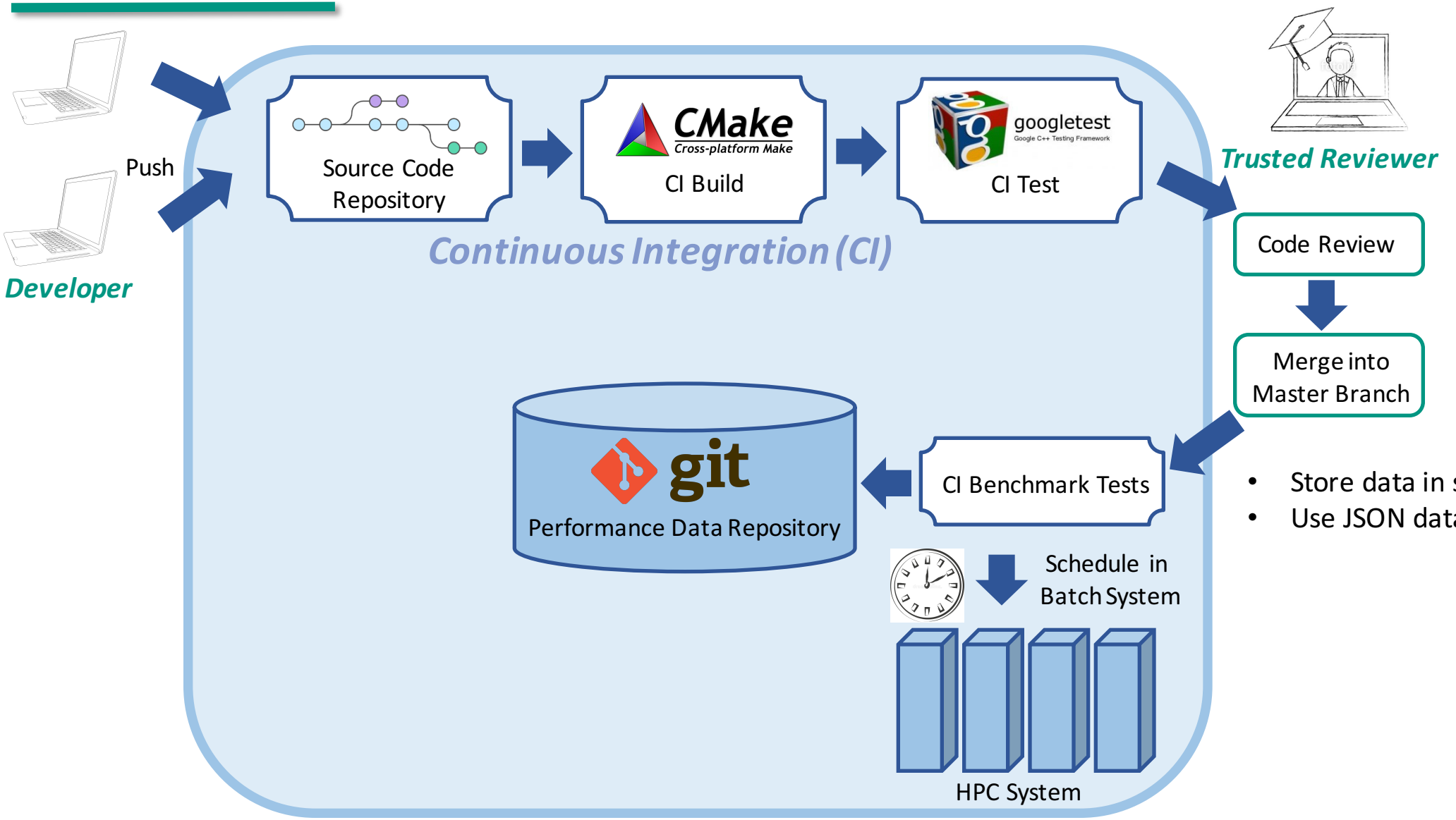
- Manual code reviews by trusted reviewers who take responsibility for executing code on the HPC system.

A Healthy Software Development Cycle



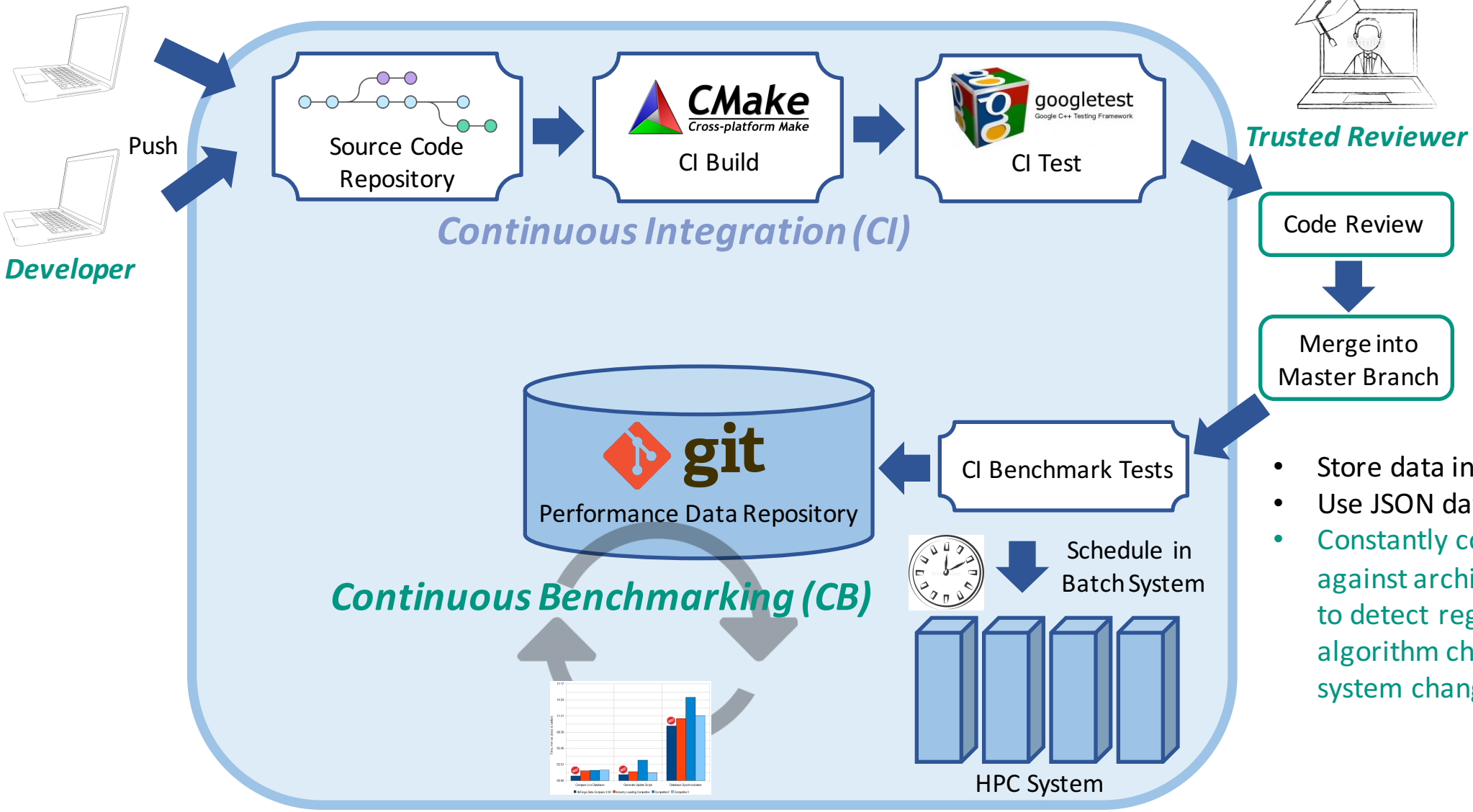
- Manual code reviews by trusted reviewers who take responsibility for executing code on the HPC system.
- How to archive the results and make them accessible?
- How do I analyze them?
- How to automatically detect performance degradations?

A Healthy Software Development Cycle



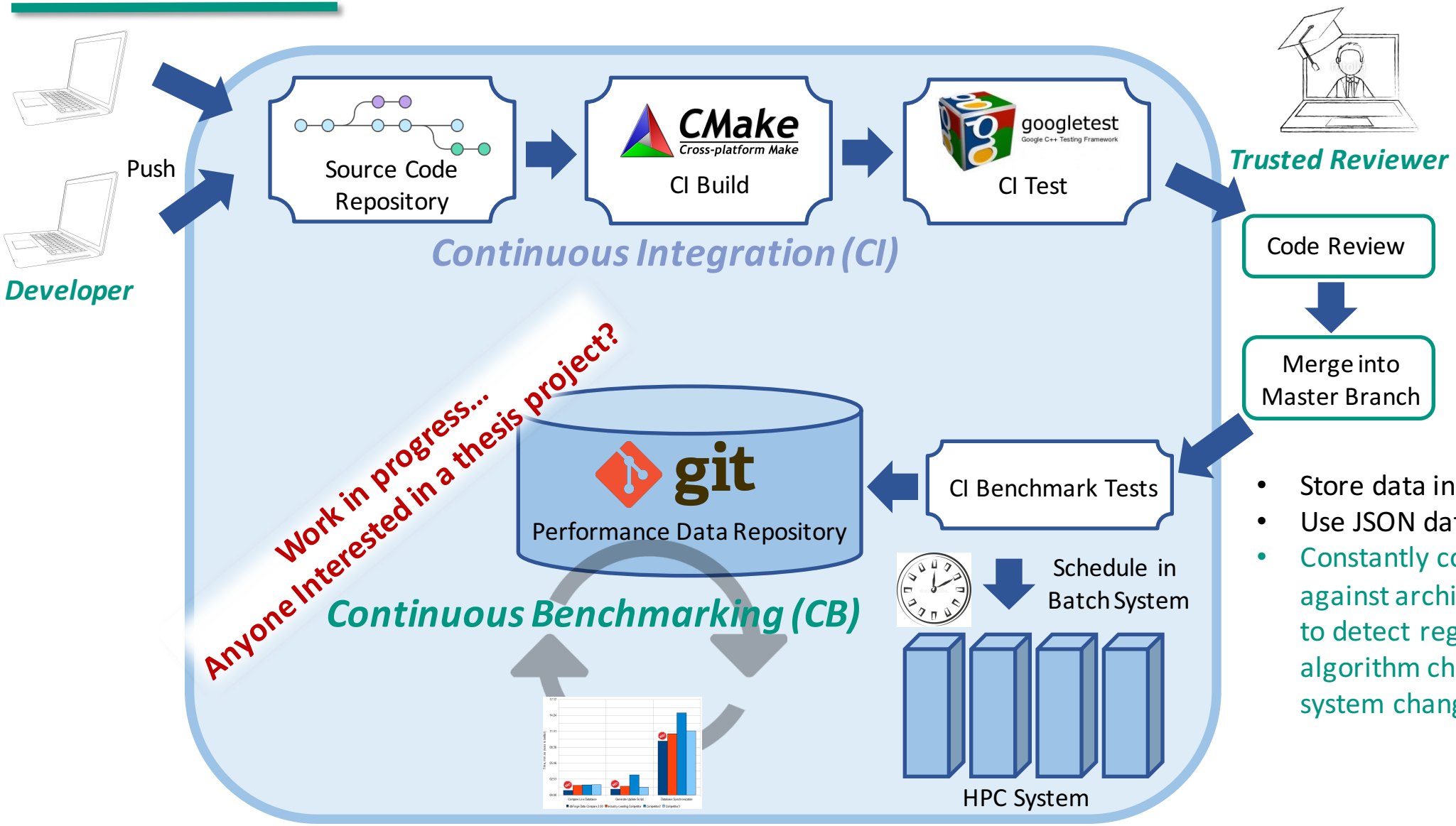
- Store data in separate repository.
- Use JSON data exchange format.

A Healthy Software Development Cycle



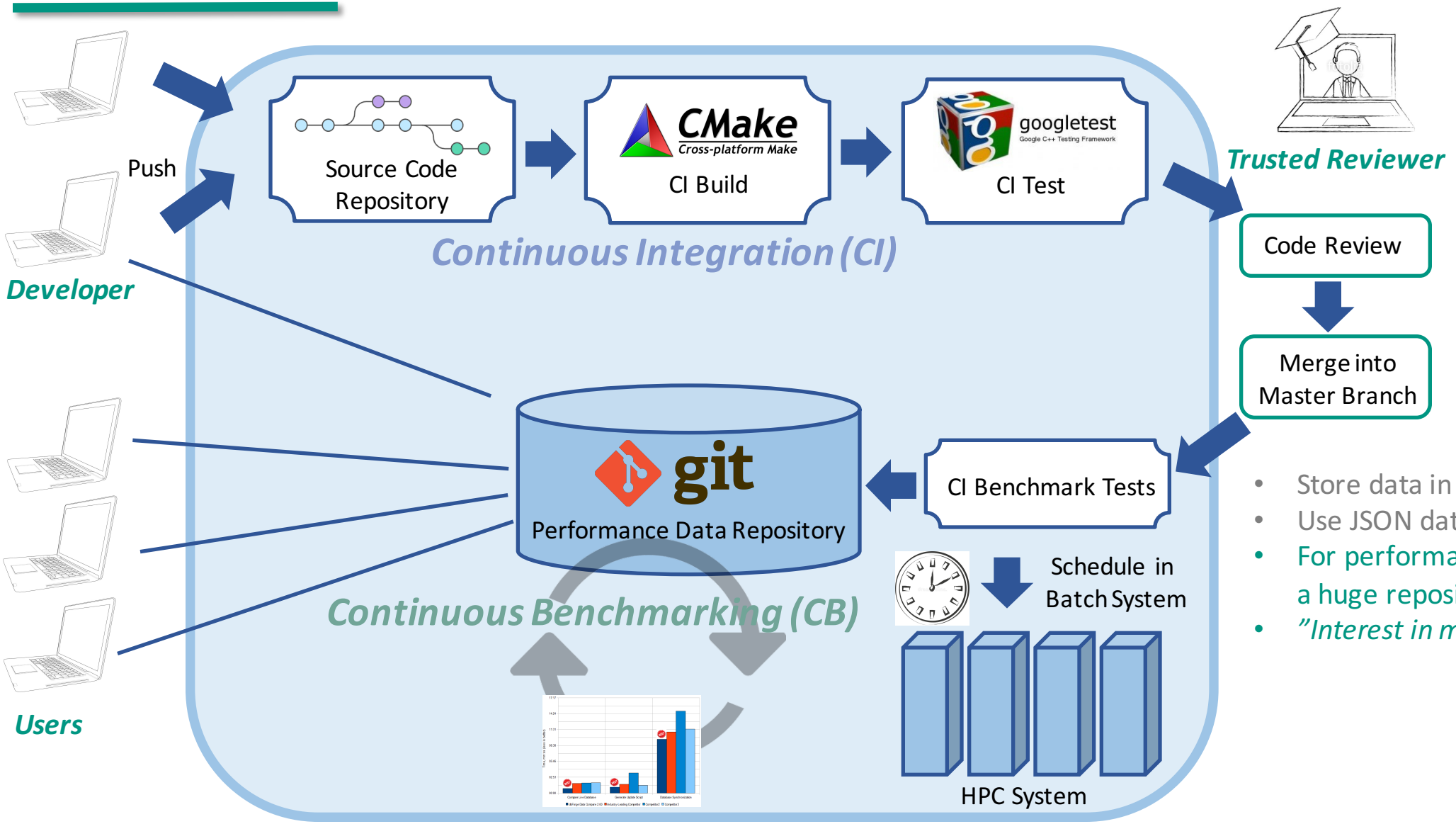
- Store data in separate repository.
- Use JSON data exchange format.
- Constantly compare performance against archived performance data to detect regressions caused by algorithm changes, compiler / OS system changes, etc.

A Healthy Software Development Cycle



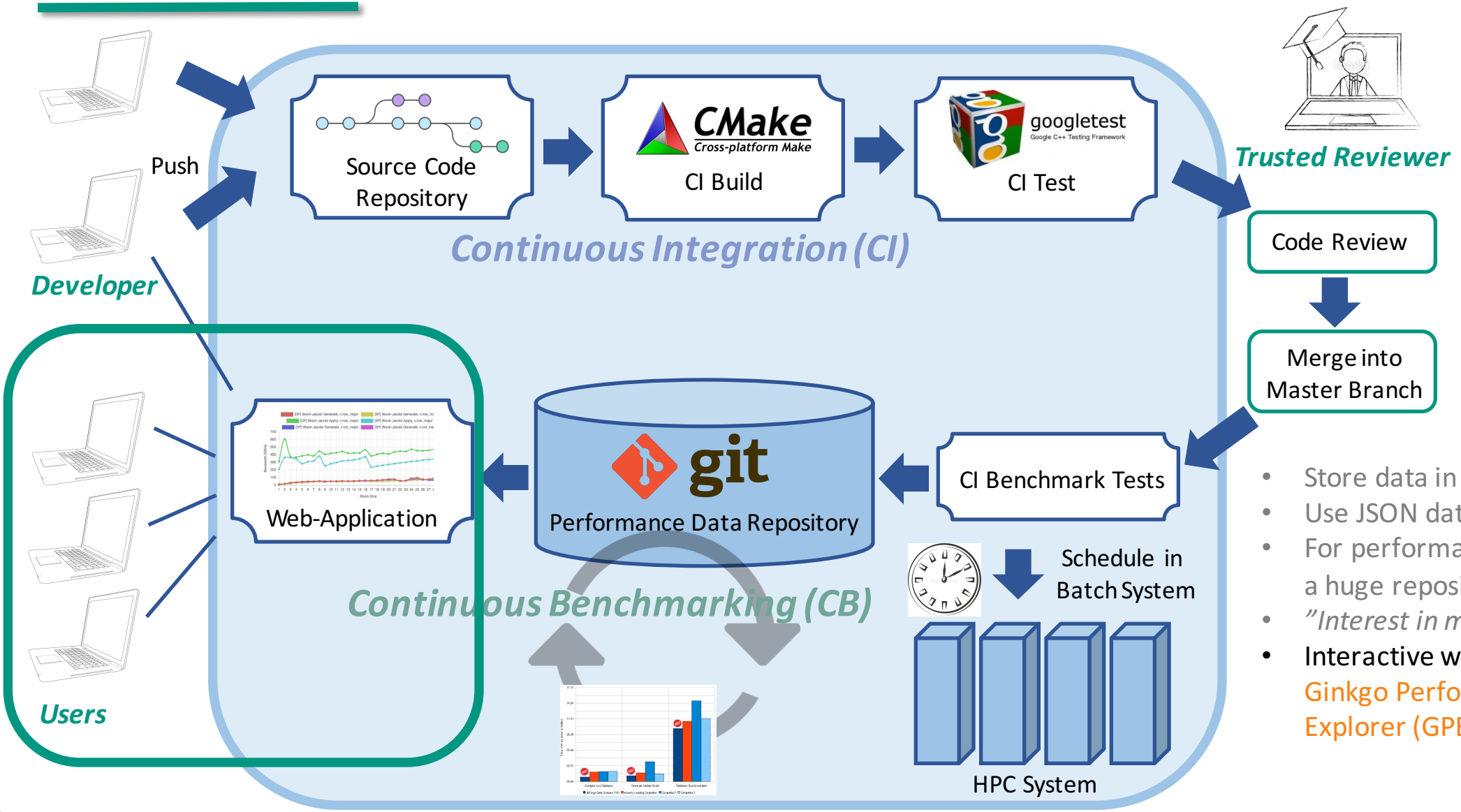
- Store data in separate repository.
- Use JSON data exchange format.
- Constantly compare performance against archived performance data to detect regressions caused by algorithm changes, compiler / OS system changes, etc.

A Healthy Software Development Cycle



- Store data in separate repository.
- Use JSON data exchange format.
- For performance analysis: Cloning a huge repository is expensive.
- *"Interest in my experiments, only."*

A Healthy Software Development Cycle



- Store data in separate repository.
- Use JSON data exchange format.
- For performance analysis: Cloning a huge repository is expensive.
- *"Interest in my experiments, only."*
- Interactive web-based Ginkgo Performance Explorer (GPE).

GPE

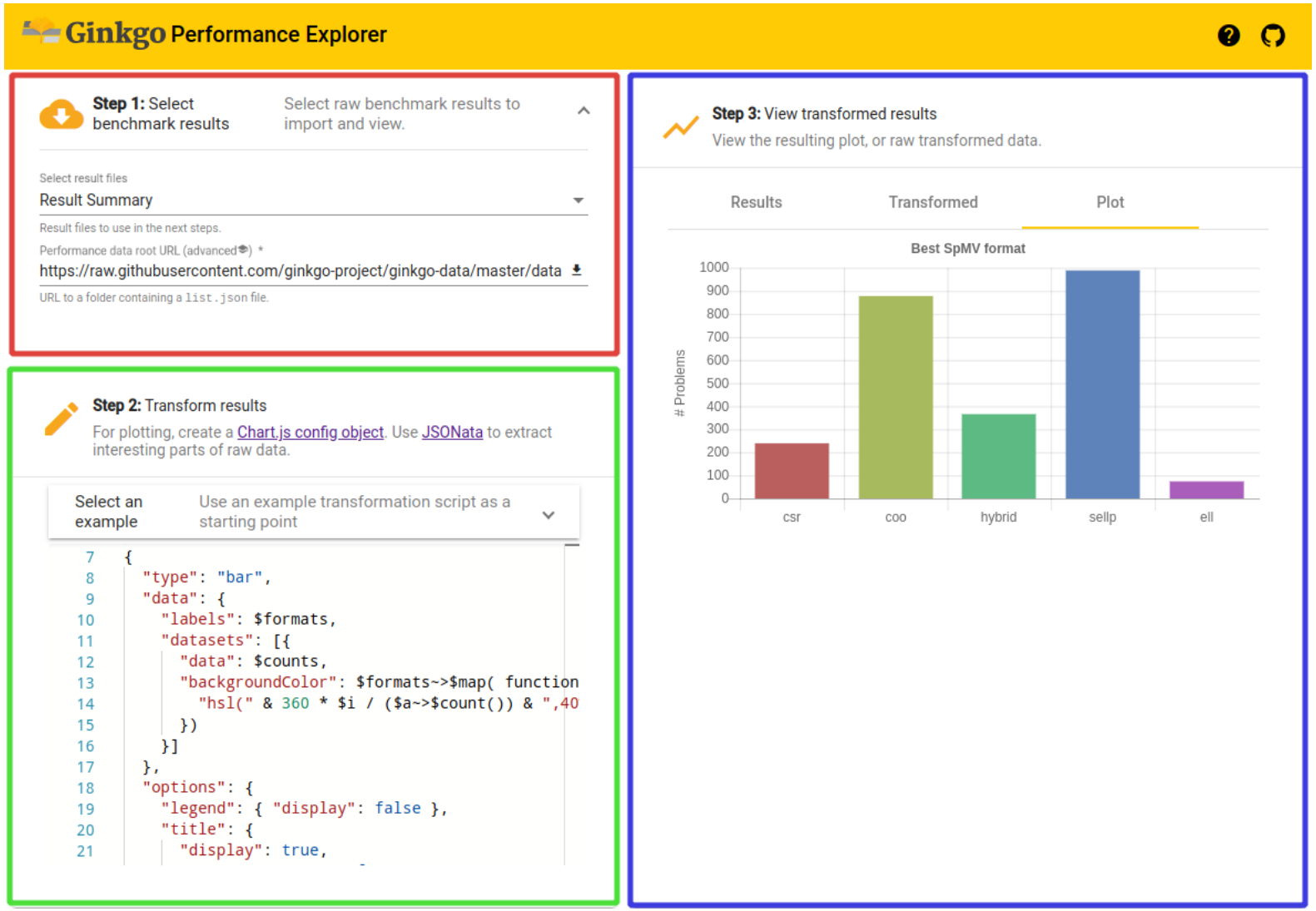
Ginkgo Performance Explorer (GPE)

Karlsruhe Institute of Technology

Data Selection Tab

Transformation Script Editor

Data and Plot Viewer



GPE

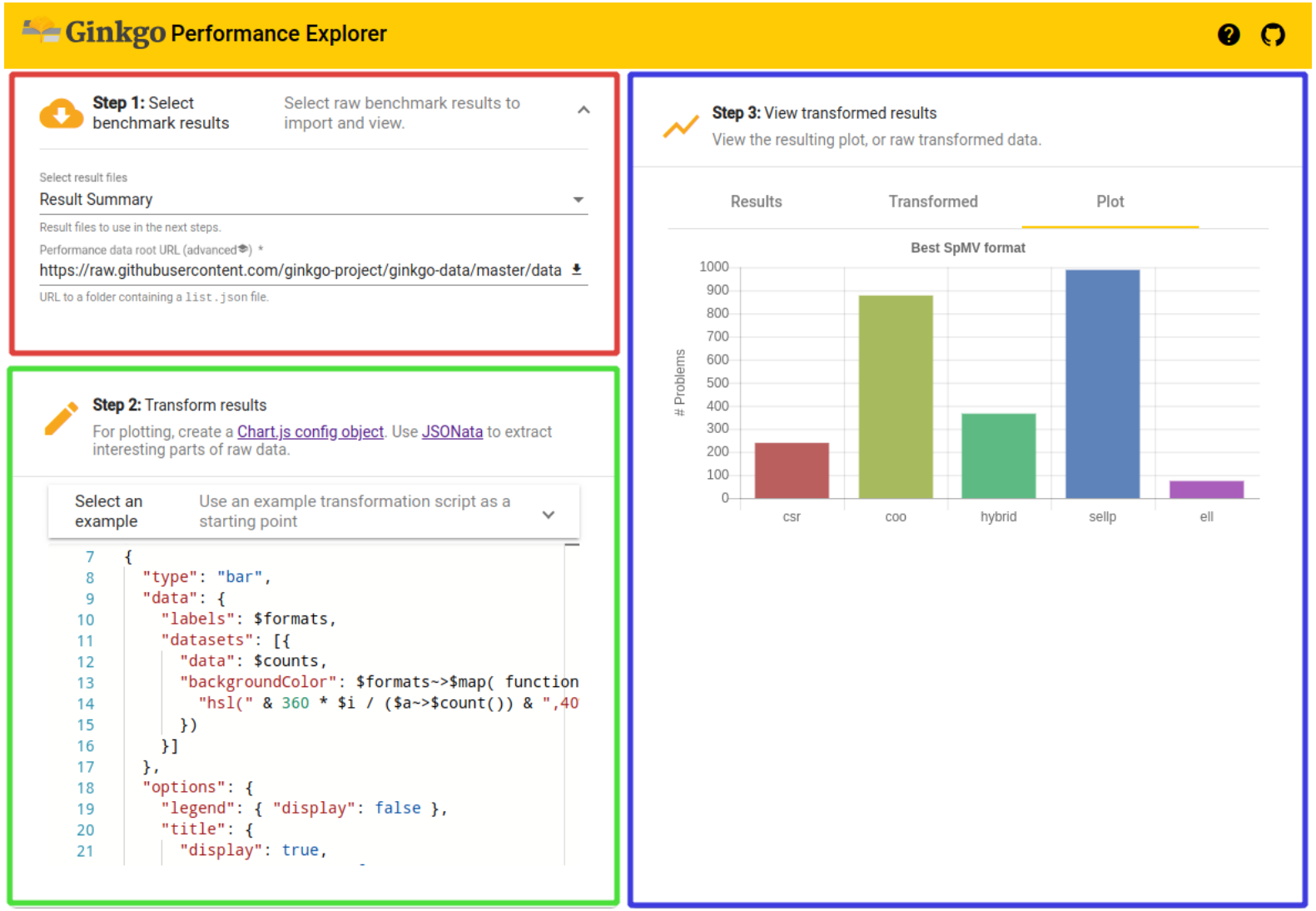
Ginkgo Performance Explorer (GPE)

Karlsruhe Institute of Technology

Data Selection Tab

1.
Select Data in Git repository.

Transformation Script Editor



Data and Plot Viewer

GPE

Ginkgo Performance Explorer (GPE)

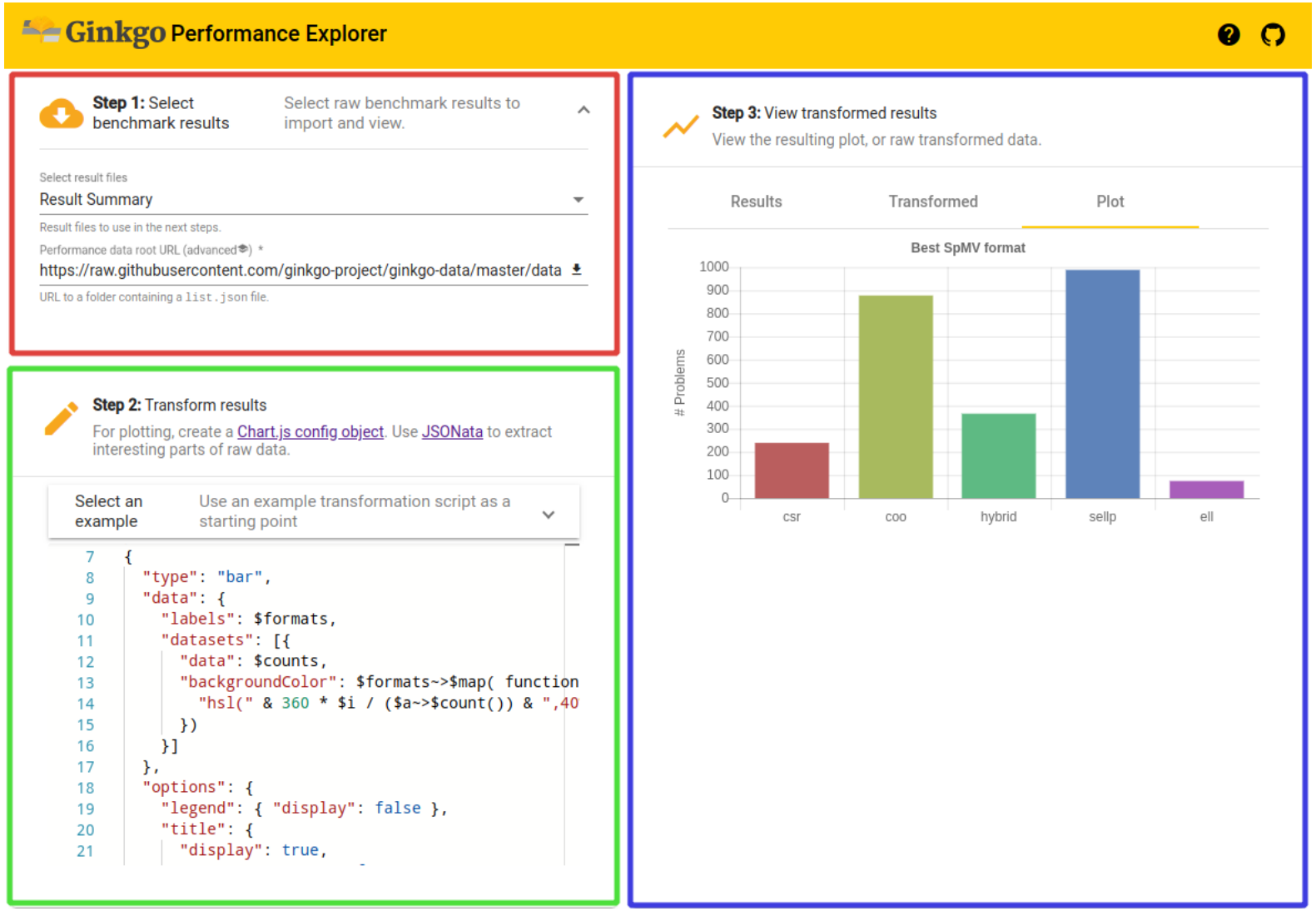
Karlsruhe Institute of Technology

Data Selection Tab

1.
Select Data in Git repository.

Transformation Script Editor

2.
Write JSONata script to visualize data (examples are provided).



Data and Plot Viewer

GPE

Ginkgo Performance Explorer (GPE)

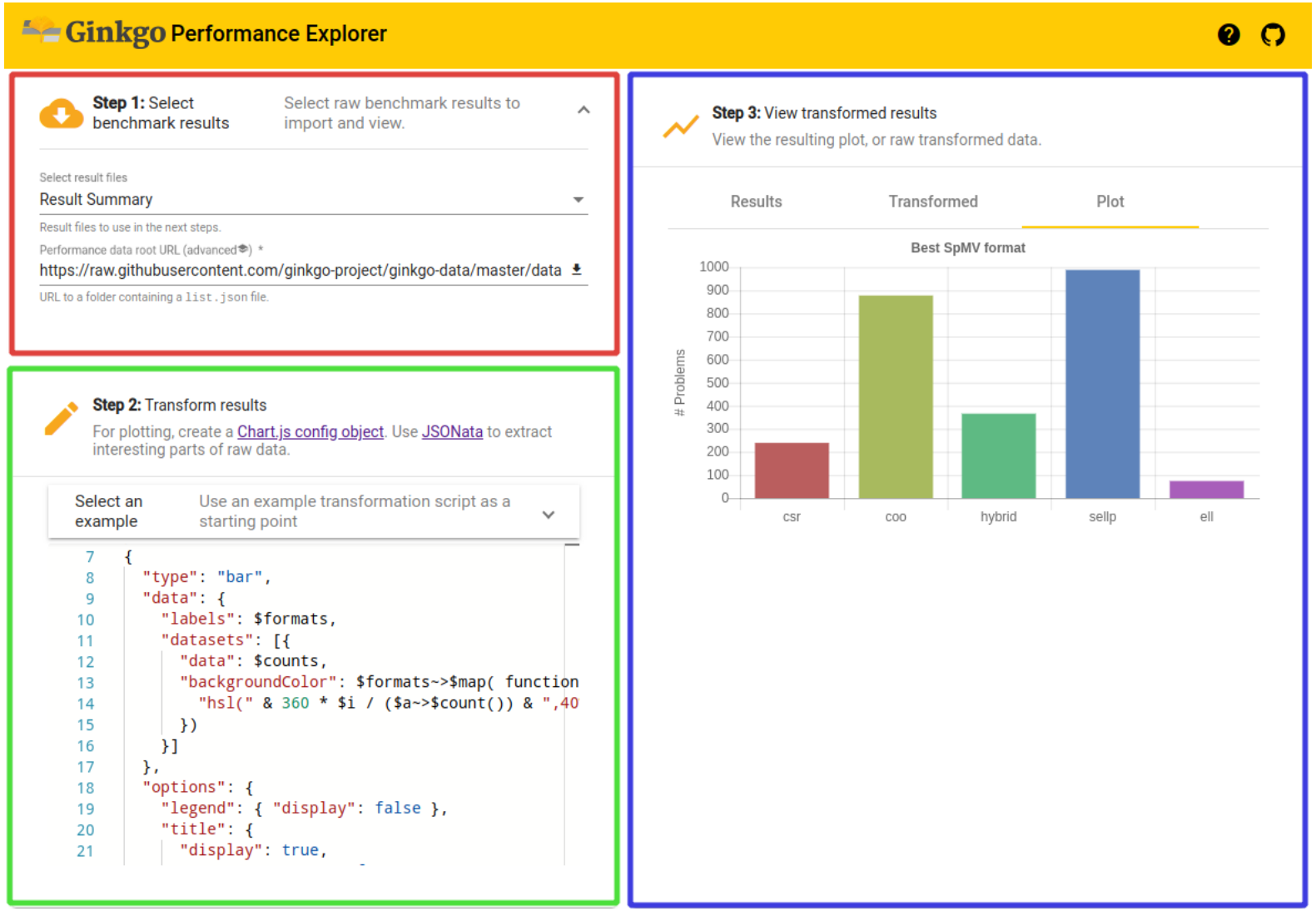
Karlsruhe Institute of Technology

Data Selection Tab

1.
Select Data in Git repository.

Transformation Script Editor

2.
Write JSONata script to visualize data (examples are provided).



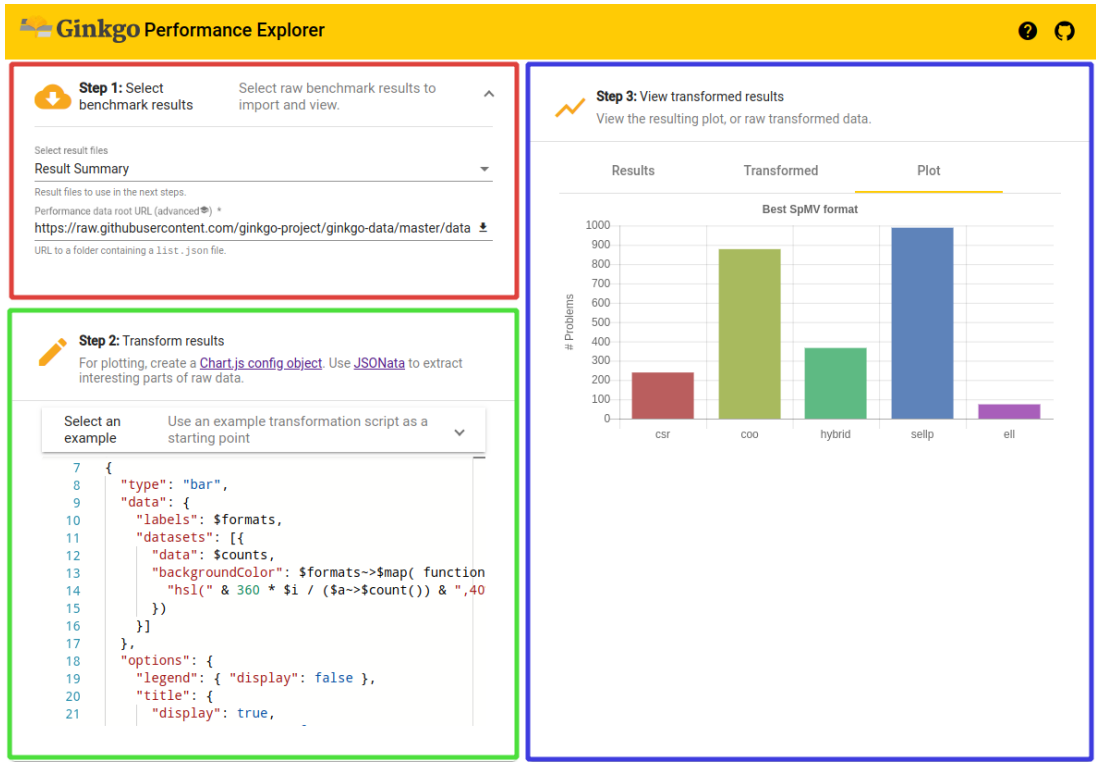
Data and Plot Viewer

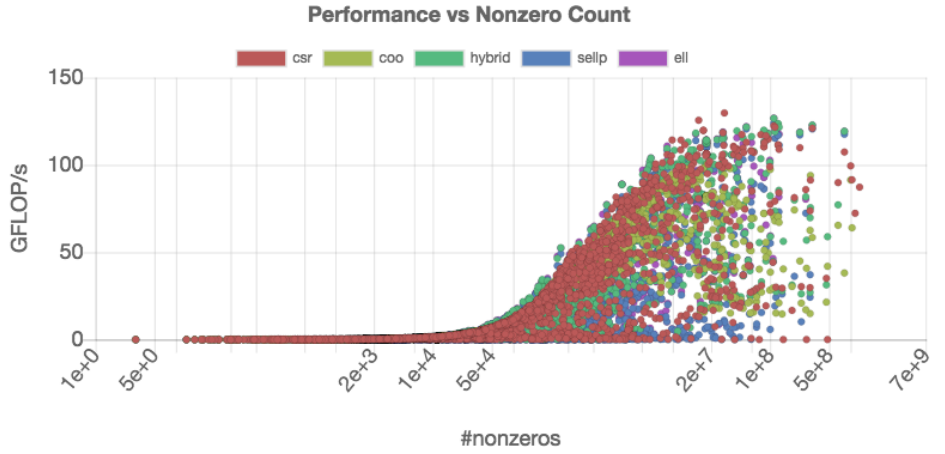
3.
Analyze data visually.

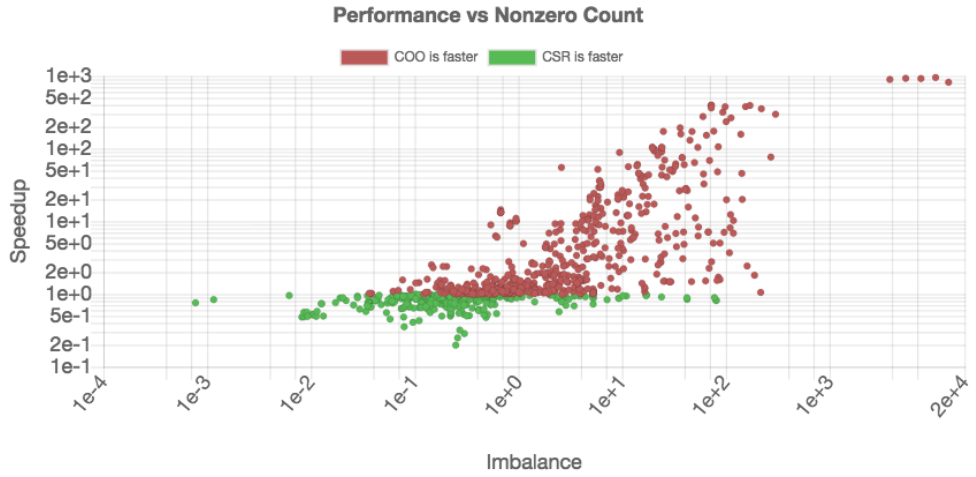
GPE

Ginkgo Performance Explorer (GPE)

Allows anyone to access & visualize Ginkgo Performance data in a browser.





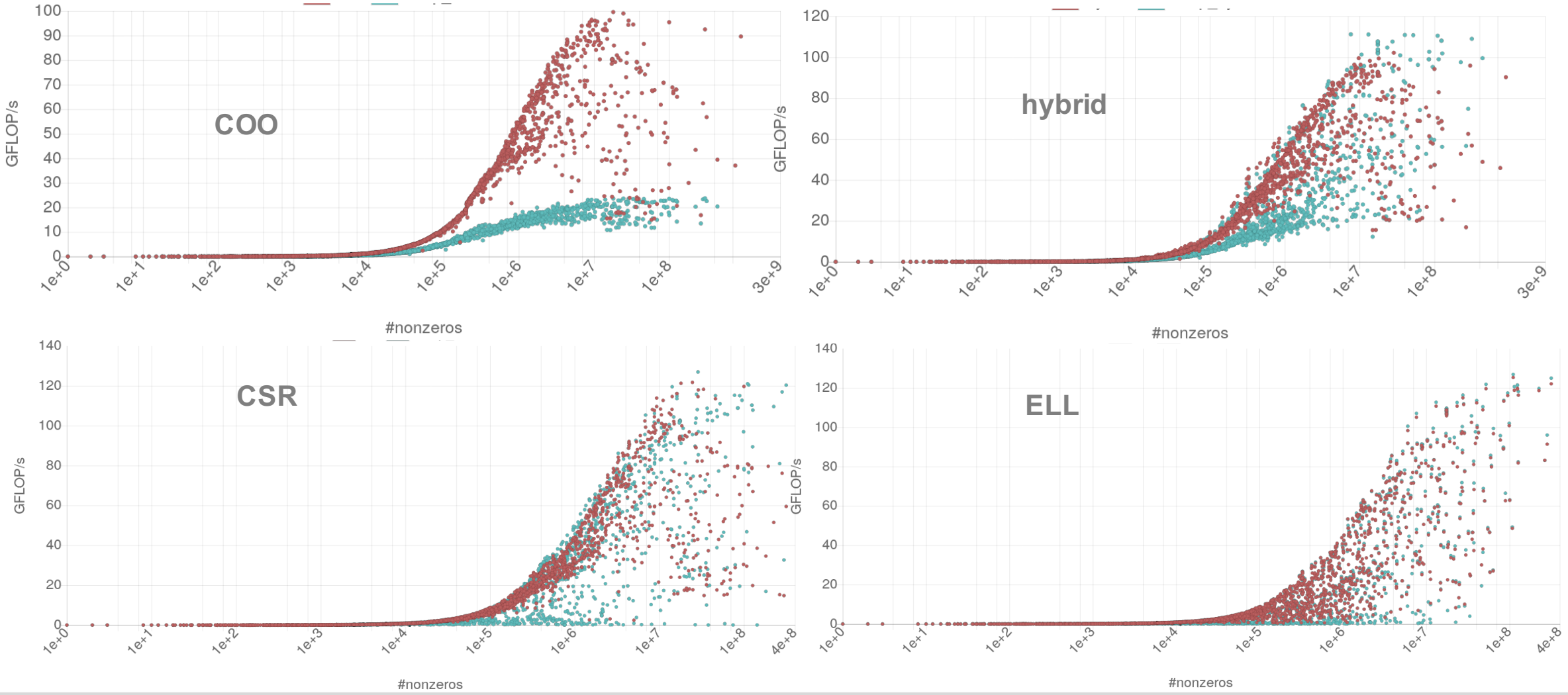


https://ginkgo-project.github.io/gpe/

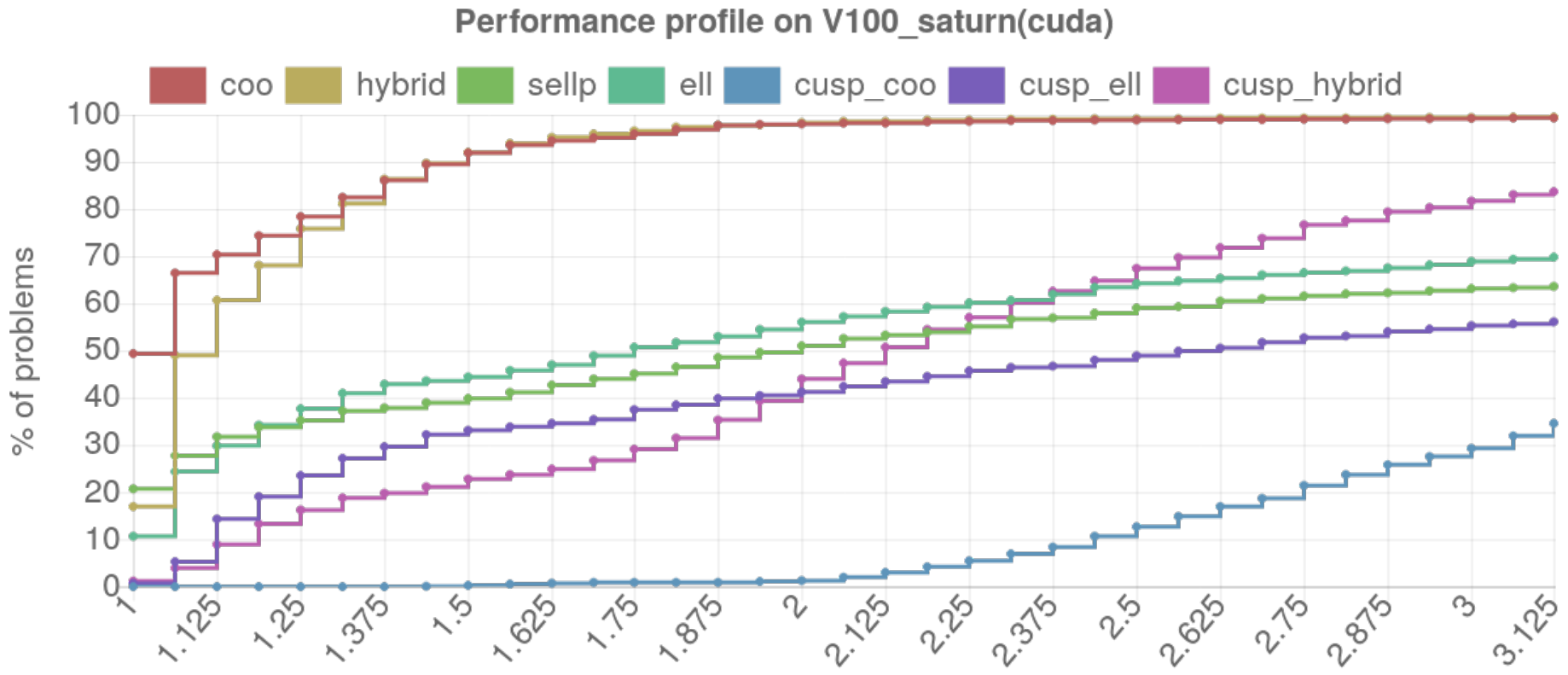
Ginkgo Performance Explorer (GPE)

Karlsruhe Institute of Technology

Data for 3,000 test matrices from SuiteSparse, sparse matrix vector product performance.



Ginkgo Performance Explorer (GPE)



Dolan & More: Benchmarking optimization software with performance profiles

GPE

Summary

Continuous Benchmarking Benefits

https://bssw.io/

- Archiving performance data along with execution parameters ensures **full benchmark reproducibility**.
- **Comparing** the performance results over the code lifetime identifies **performance degradations**.
- **Ease of use**: the setup allows to launch benchmark with few clicks.

Summary

Continuous Benchmarking Benefits

https://bssw.io/

- Archiving performance data along with execution parameters ensures **full benchmark reproducibility**.
- **Comparing** the performance results over the code lifetime identifies **performance degradations**.
- **Ease of use**: the setup allows to launch benchmark with few clicks.

Ginkgo Performance Explorer (GPE) Benefits

- The design of GPE efficiently realizes the analysis as **web service**, removing the need for downloading performance data to local disk or installing additional software.
- **External developers** without access to HPC systems can test and engineer **their codes on HPC resources**.
- **Extensibility**: Option to **compare** performance with **other software** libraries.

GPE

https://bssw.io/blog_posts/the-art-of-writing-scientific-software-in-an-academic-environment

Anzt et al: **"Towards Continuous Benchmarking: An Automated Performance Evaluation Framework for High Performance Software"**, PASC 2019.

Summary

Work in progress...
Anyone Interested in a thesis project?

Continuous Benchmarking Benefits

- Saving performance data along with execution parameters ensures **full benchmark reproducibility**.
- **Comparing** the performance results over the code lifetime identifies **performance degradations**.
- **Ease of use**: the setup allows to launch benchmark with few clicks.

https://bssw.io/

Ginkgo Performance Explorer (GPE) Benefits

- The design of GPE efficiently realizes the analysis as **web service**, removing the need for downloading performance data to local disk or installing additional software.
- **External developers** without access to HPC systems can test and engineer **their codes on HPC resources**.
- **Extensibility**: Option to **compare** performance with **other software** libraries.

GPE

https://bssw.io/blog_posts/the-art-of-writing-scientific-software-in-an-academic-environment

Anzt et al: **"Towards Continuous Benchmarking: An Automated Performance Evaluation Framework for High Performance Software"**, PASC 2019.

Learn More about Ginkgo

- Open-source C++ framework for sparse linear algebra.
- Sparse linear solvers, preconditioners, SpMV etc.
- Generic algorithm implementation:
  + reference kernels for checking correctness;
  + architecture-specific highly optimized kernels.
- Focused on GPU accelerators (i.e. NVIDIA GPUs).
- Software quality and sustainability efforts guided by xSDK community policies:
  - open source, BSD
  - Doxygen
  - CMake Cross-platform Make
  - googletest Google C++ Testing Framework, GitLab CI

https://xsdk.info/

Ginkgo

https://ginkgo-project.github.io/


Terry Cojean


Goran Flegar


Thomas Grützmacher


Pratik Nayak


Tobias Ribizel

better scientific software

https://bssw.io/