VI. Recent Developments & Future Trends in Trilinos

14 TRILINOS DevOps Pipeline (TDOP) Planning

15 Introduction of New Strategic and Operational Leadership

16 TRILINOS Framework Updates

17 TRILINOS Core Product Area Update

18 TRILINOS Solver Updates

19 TRILINOS Discretizations and Analysis Product Update

What to expect?

Trilinos updates:

- Updates from the **Trilinos leadership** as well as for the **DevOps pipeline** and the **different product areas**:
  - What has changed / been added recently?
  - Work-in-progress
  - Future trends and plans for TRILINOS
- Selected aspects are being discussed (non-exhaustive!)

Breaking change in 2024 & 2025:

- Deprecation of the EPETRA stack

Later today: Hackathon

To help with the transition to TPETRA, we will have a hackathon w/ TPETRA developers. Ask your questions. Work on your code. Together.

Disclaimer

The following slides are based on the slide decks of presentations given by *Curt Ober*, *Sam Browne*, *Roger Pawlowksi*, *Christian Glusa*, and *Mauro Perego* at the TRILINOS User Group Meeting 2023 (TUG '23); please see the website for more details.

Trilinos DevOps Pipeline (TDOP) Planning

Trilinos Philosophy

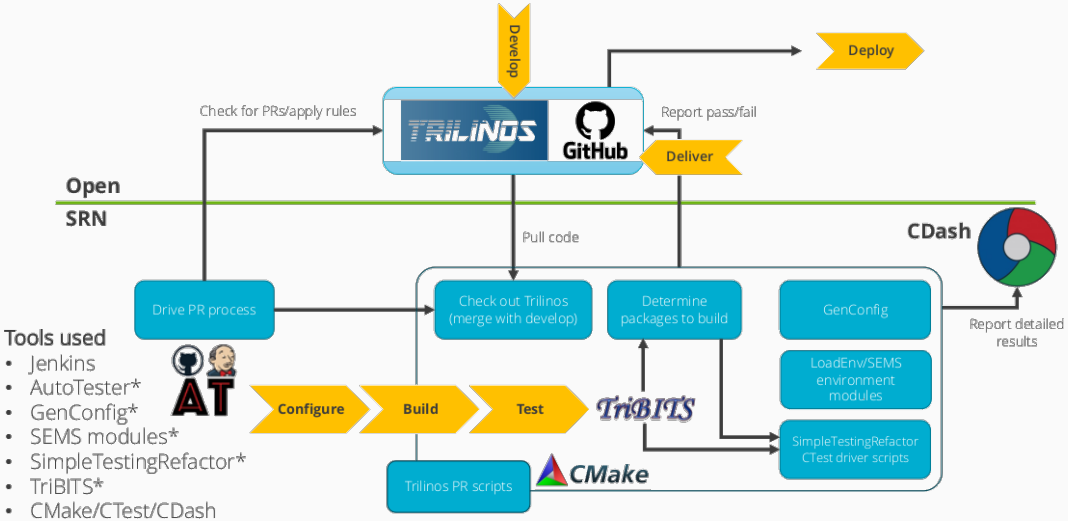
> *"The* Trilinos *Project is an effort to facilitate the design, development, integration and ongoing support of mathematical software libraries within an object-oriented framework for the solution of large-scale, complex multi-physics engineering and scientific problems.* Trilinos *addresses two fundamental issues of developing software for these problems: (i)* ***Providing a streamlined process and set of tools for development of new algorithmic implementations*** *and (ii)* ***promoting interoperability of independently developed software****."*

Heroux et al., "An Overview of the Trilinos *Project", ACM Transactions on Mathematical Software, Vol. V, No. N, December 2004, Pages 1–27*

> *"The* Trilinos *project was established to address two important needs: (1) bringing teams of library developers together in order to* ***leverage commonalities and produce compatible software components****, formally called packages and (2) to* ***amortize the cost and efforts associated with more formal software engineering requirements****. With a modest level of coordination and without unduly compromising package team autonomy,* Trilinos *project members could leverage each other's efforts, consolidate commonly needed tools, make packages compatible, and define a common set of software engineering tools and processes.."*

Heroux and Willenbring, "A new overview of the Trilinos *project", Scientific Programming 20 (2012), p. 83–88*

Overview of Current Trilinos DevOps Pipeline



Executive Summary – Trilinos Plans to . . .

DEVELOP

- Remain single repository to maintain developer productivity
- Retain key capabilities of TriBITS and form a support team

CONFIGURE

- Utilize ASC DevOps common Trilinos configurations (e.g., RAMSES and CompSim)
- Provide/maintain a Spack recipe that others can use (e.g., ASC stakeholders and Spack)

BUILD

- Maintain/support CMake & TriBITS and Spack builds
- Incorporate Containers and GitHub Actions to catch build errors and keep builds clean

TEST

- Add Integration testing for Trilinos packages (e.g., Kokkos and KokkosKernels)
- Support application's integration testing of Trilinos to mitigate integration issues

DELIVER & DEPLOY

- Support both delivery (Trilinos GitHub) and deployment (Spack)
- Steward Trilinos's Spack recipe with support from Framework and Trilinos developers

Trilinos remains a single repository, retaining the capabilities of TriBITS

- What are the **complaints and perceived issues**?
  - *"*Trilinos *is too big.* Trilinos *is too complex."* → Too many packages/dependencies
  - Should Trilinos be *"partitioned"*?
  - Why do we have TriBITS?
- **Points to consider**
  - Survey says . . . 76% favored single Trilinos repository
    - Large loss of productivity with multiple repos for developers and applications
  - TriBITS provides CMake layer of common/consistent functionality across Trilinos packages
    - Most large software projects have some CMake layer (e.g., LNLL's BLT and VTK modules)
  - Recent and ongoing TriBITS improvements (e.g., modern CMake and external builds)
- **Planned solution/response**
  - **Single Trilinos repository** to maintain developer productivity
  - Retain **key capabilities of TriBITS** and form a **support team**
  - **Review packages/dependencies** to ensure we have the **minimum set**
  - Consider **not using (and removing) subpackages capability** of TriBITS/Trilinos

Simplify Trilinos configurations and provide Spack recipe

- What are the **complaints and perceived issues**?
  - Too many configuration options to get the build you would like
  - Not easy to configure/build for a single/few package(s)
- **Points to consider**
  - Survey says . . .
    - Configuration preference - CMake/TriBITS (35%), CMake/Scripting (32%), CMake/Package Manager/Scripting (23%)
    - **Documentation on configuring/building needs improvement. New users have difficulty.**
- **Planned solution/response**
  - Improve documentation for configuration options and provide examples
  - Provide **"recommended" configure defaults** based on PR testing
  - Utilize ASC DevOps common Trilinos configurations (e.g., RAMSES and CompSim)
  - Review dependencies to **ensure required and optional dependencies are correctly defined**
  - **Provide/maintain a Spack recipe that others can use** (e.g., ASC stakeholders and Spack)
  - Generate intra-package/TPL dependencies with TriBITS for CMake & TriBITS and Spack recipe

Maintain CMake+TriBITS and Spack builds, incorporate Containers and GitHub Actions

- What are the **complaints and perceived issues**?
  - Build times are long
- **Points to consider**
  - Survey says ...
    - Trilinos build times are not long compared to application build times. (21 to 9 respondents)
    - **>90% build times less than an hour**
  - Container usage is dependent on CEE LAN RHEL 8
- **Planned solution/response**
  - Continue CMake & TriBITS builds (more for developers/power users)
  - Expand support for Spack & CMake builds (more for stakeholders/future usage/sysadmins)
  - Add a **Spack PR build** (meaning a Spack build must pass to commit)
  - Incorporate Containers and GitHub Actions to catch build errors and keep builds clean
  - Utilize **build caches (ccache) to increase speed**

Add integration testing for Trilinos and support integration testing of Trilinos

- What are the **complaints and perceived issues**?
  - **PR testing results not visible to foreign nationals and external developers**
  - Trilinos needs **better performance testing**
  - Lengthy process to snapshot packages like Kokkos → integration takes too long
- **Points to consider**
  - Trilinos testing covers CTS/testbed (SRN) platforms
  - There are gaps in PR testing due to **hardware availability** (e.g., AMD GPUs)
  - Integration testing between
    - Trilinos packages would introduce additional overhead
    - Upstream packages/Trilinos/applications can reduce turn-around time
  - Trilinos has **internal "performance" dashboard**
- **Planned solution/response**
  - Add integration testing for Trilinos packages (e.g., Kokkos and KokkosKernels)
  - Support application's integration testing of Trilinos to mitigate integration issues
  - **Add application use cases for performance monitoring**
  - Expand performance testing and memory monitoring (e.g., Watchr)
  - Move PR and nightlies to SON/OHPC for foreign nationals and external collaborators

Support both Trilinos GitHub and Spack and steward Trilinos's Spack recipe

- What are the **complaints and perceived issues**?
  - "Need to deliver code changes more quickly to applications."
  - Difficult to use Spack's recipe for Trilinos
- **Points to consider**
  - Survey says ...
    - **Preference is to use CMake installation over package manager** (20 to 6 respondents)
    - **Package manager (e.g., Spack) is path forward** for the future
  - In the past, primarily only delivered Trilinos and applications deployed.
- **Planned solution/response**
  - Support both delivery (Trilinos GitHub) and deployment (Spack)
  - Steward Trilinos's Spack recipe with support from Framework and Trilinos developers

Next Steps . . .

FY24

- Epetra → Tpetra Transition for Trilinos packages
- Current planned tasks
  - Test/support CompSim and RAMSES Trilinos configurations (coordinating with ASC DevOps)
  - **Ownership/support of Spack-based Trilinos build added to nightly/PR testing**
  - **Transition from SEMS Autotester to GitHub Actions & Containers**
  - **Transition from GenConfig to Spack for configuration management**
- Continued planning of TDOP

FY25

- Epetra → Tpetra transition for stakeholders
- Start other planned TDOP responses

Introduction of New Strategic and Operational Leadership

New Leadership Structure

Strategic Leadership

- Should cover broad research areas
  - Algorithms research
  - Big picture – vision for possible new areas
  - Software Sustainability
  - Long-term application needs
- Current Strategic Leaders
  - *Mike Heroux*
  - *Eric Phipps*
  - *Siva Rajamanickam*
  - *Heidi Thornquist*
  - *Jim Willenbring*





Operational Leadership

- Should cover operational areas
  - Product manager
  - Framework lead
  - New area leads
    - Representation for every package
    - Need to balance packages/effort
- Current Operational Leaders
  - *Sam Browne* – **Framework**
  - *Christian Glusa* – **Solvers**
  - *Curt Ober* – **Product manager**
  - *Roger Pawlowski* – **Trilinos Core**
  - *Mauro Perego* – **Discretizations and Analysis**





Current Trilinos Product Areas and Packages

Framework	Data Services	Discretizations	Linear Solvers	Nonlinear Solvers	Product Manager
• PyTrilinos • Teuchos • TrilinosCouplings • ~~TriUtils~~	• ~~Epetra~~ • ~~EpetraExt~~ • Galeri • ~~Isorropia~~ • Kokkos • Kokkos Kernels • Pamgen • SEACAS • Tpetra • Xpetra • Zoltan/Zoltan2	• Compadre • ~~Domi~~ • ~~FEI~~ • ~~Intrepid~~/Intrepid 2 • MiniTensor • Panzer • Percept • Phalanx • Shards • Krino • STK	• Adelus • ~~Amesos~~/Amesos2 • Anasazi • ~~AztecOO~~ • Belos • ~~Ifpack~~/Ifpack2 • ~~Komplex~~ • ~~ML~~ • MueLu • Pliris • ShyLU/ShyLU-DD/ShyLU-Node • Stratimikos • Teko	• ~~Moertel~~ • NOX • ~~Pike~~ • Piro • ROL • RTOp • ~~Rythmos~~ • Sacado • Stokhos • Tempus • Thyra	• N/A

• ~~Package~~ – Slated for Archival in FY23-FY25

Package – Snapshotted package

Proposed Trilinos Areas and Packages

Framework	Core	Solvers	Discretizations and Analysis	Product Manager
• N/A	• Galeri • Kokkos • Kokkos Kernels • Pamgen • SEACAS • Tpetra • Zoltan/Zoltan2 • PyTrilinos • Teuchos • RTOp • Thyra	• Adelus • Amesos2 • Anasazi • Belos • Ifpack2 • MueLu • Pliris • ShyLU/ShyLU-DD/ShyLU-Node • Stratimikos • Teko • NOX • Xpetra	• Compadre • Intrepid2 • MiniTensor • Panzer • Percept • Phalanx • Shards • Krino • STK • Sacado • Stokhos • Tempus • TrilinosCouplings • Piro • ROL	• N/A

Trilinos Framework Updates

Current Trilinos Product Areas and Packages

Selected updates (FY23)

- **7 TRILINOS packages deprecated and removed from repository**
- PR builds
  - CCACHE tooling to accelerate PR builds
  - Expanded warning flags
  - Turned off EPETRA and other packages in CUDA build
- Nightly builds
  - Added Clang & OPENMP, C++20 builds
  - Added KOKKOS/KOKKOSKERNELS develop -> TRILINOS develop build

Selected plans for FY24

- Promote C++20 build to PR status
- **Add at least one Spack-based PR build**
- **Migrate from SEMS Autotester to GitHub Actions for CI/PR builds**
- Stretch goal: use **Spack to manage configurations** of TRILINOS for PR testing (as opposed to current GenConfig system)
- **Add advanced hardware builds** as hardware becomes available (e.g., AMD GPU hardware)

Trilinos Core Product Area Update

Teuchos updates

- Mostly in **maintenance mode**
- **Improved YAML parser support**
- TRILINOS leadership: discussions on what we can clean up and remove

Tpetra updates

- FY23
  - Performance, Performance Testing/Monitoring, software quality improvements
  - **Epetra → Tpetra transition**
- FY24
  - **Application transition support: Epetra → Tpetra**
  - Cleanup unnecessary D2H (device-to-host) and H2D (host-to-device) transfers

Epetra depracation

- **Deprecation of Epetra from Trilinos (End of FY24; Deadline Sept. 2024)**
  - All Trilinos packages compile and function without Epetra
    - Nightly testing without Epetra ✓
  - **All "needed" Epetra testing has equivalent Tpetra testing**
  - Packages consult with stakeholders to determine any missing Tpetra functionality
    - Try to assess performance impacts
  - **Epetra still available and tested during FY24**
- **Deprecation of Epetra from stakeholder applications (End of FY25; Deadline Sept. 2025)**
  - Applications transition to Tpetra
  - Packages handle any new issues and performance problems
- **Epetra stack archival to separate repository (Beginning of FY25; Oct. 2025)**

Zoltan2 updates

- Two new graph partitioners in (coming to) Zoltan2:
  - **Sphynx**: Spectral partitioner, multi-GPU
    - Speedup from randomized eigensolver
  - **Jet**: Multilevel, KOKKOS-based partitioner (CPU and GPU)
    - High quality. Beats Metis by 6-10% in edge cuts. Coming soon.
    - Currently limited to single GPU, multi-GPU in progress.

See the TUG' 23 slides for more details

PyTrilinos2 updates

- **New package**
- Auto-generation of Python interface using PyBind11 and binder

PyTrilinos updates

- Plan to deprecate and remove due to PYTRILINOS2 (no timeline at this point)
  PyTrilinos implementation based on SWIG wrappers

RTOp, Thyra, and Xpetra updates

- **Maintenance mode**
- Leadership: Is XPETRA a potential candidate for deprecation with EPETRA removal?

Kokkos-related update

- TRILINOS can now build TRILINOS against an external install of Kokkos
  - TRILINOS SPACK builds do this by default

See KOKKOS *talks (Monday) and TUG' 23 for more details*

Trilinos Solver Updates

Many algorithmic developments

Amesos2 update

- Updates to STRUMPACK, SuperLU_dist, MUMPS and LAPACK adapters

Belos & Anasazi update

• Randomized eigensolver	*Switzer, Boman, Loe*
• Improved testing with Tpetra	*Thornquist, NGA*
• Extend GCRO-DR linear solver to use Kokkos linear algebra for GEMMA	*Dang, Loe*
• **(WIP)** Integration of serial dense matrix traits in Belos to enable use of Teuchos/Kokkos objects	*Loe, Thornquist*

Ifpack2 update

- 4th kind Chebyshev smoother — *Phillips*
- Algorithmic and performance improvements for BlockTriDiag and BlockJacobi — *Liegeois*

  (WIP) Schur complement approach
- Stream based RILU(k) and triangular solves — *Dang*
- Optimize Kokkos Kernels MDF ILU(0) solver and expose it in IFPACK2 — *Ransegnola*
- Patch solver with data compression — *Harper*
- **(Upcoming)** block version of traditional ILU(0) — *Foucar*

MueLu update

• Better ML/MueLu compatibility (parameter translation, aggregation algorithms, ..)	*Siefert, Tuminaro*
• Reitzinger-Schöberl type multigrid for Maxwell problems	*Siefert, Tuminaro*
• BlockCRS support	Siefert
• Improved setup performance on device (TAFC Tpetra changes)	*Berger-Vergiat, Hu, Ren*
• Reformulated Darcy solver	*Glusa*
• Matrix-free multigrid with user-specified operators	*Glusa*
• AMG for hierarchical matrices	*Glusa*
• **(WIP)** Refactor of host-only and Kokkos code paths	*Glusa, Harper*
• **(WIP)** MueLu tutorial overhaul	*Harper, Mayr*
• **(Upcoming)** Matrix-free AMG	*Harper*

NOX update

- Refactored internal use of model evaluators — *Ober, Pawlowski*
- LOCA Householder constraint solver can now be nested within a Tempus transient problem (Tpetra version) — *Pawlowski*

ShyLU update

- FastILU algorithmic improvements and testing, new block version — *Foucar*
- Transpose solve with Basker — *Ellingwood*
- Tacho: runs with HIP. WIP: performance for solves on streams — *Yamazaki*

ShyLU/FROSch update

• Fully recursive multi-level implementation	*Heinlein, Röver*
• Monolithic coarse spaces via partition-of-unity approach	*Heinlein, Knepper, Saßmannshausen*
• GPU capabilities	*Heinlein, Yamazaki*
• **(WIP)** Spectral coarse spaces	*Heinlein, Knepper, Yamazaki*

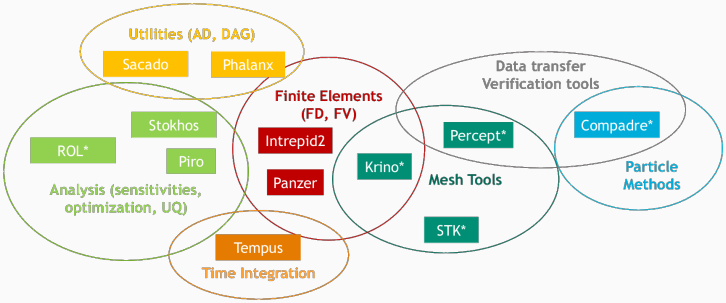
Stratimikos update

• Use of half precision preconditioners (IFPACK2, MUELU, SHYLU/FROSCH)	*Glusa, Loe, Yamazaki*

Teko update

• EPETRA dependency is now optional	NGA, Pawlowski

Trilinos Discretizations and Analysis Product Update



Epetra archival

All the packages can be built without Epetra

- **Good test coverage without Epetra**
- Some Epetra tests still need to be converted to Tpetra

Intrepid archival

- Working on transition of Krino and Percept to Intrepid2
- ROL still need to transition to Intrepid2 (ROL uses Intrepid for testing)

New effort

Develop portable tools for **efficient computation** of operator actions in a **matrix-free** fashion for **high-order finite-element discretization on unstructured meshes**.

Thank you for your attention!

Questions?