SANDIA REPORT
SAND2010-5903
Unlimited Release
Printed May 2013

Trilinos Developers SQE Guide: ASC Software Quality Engineering Practices Version 3.0

James M. Willenbring
Michael A. Heroux

Prepared by
Sandia National Laboratories
Albuquerque, New Mexico 87185 and Livermore, California 94550

Sandia National Laboratories is a multi-program laboratory managed and operated by Sandia Corporation, a wholly owned subsidiary of Lockheed Martin Corporation, for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-AC04-94AL85000.

Issued by Sandia National Laboratories, operated for the United States Department of Energy by Sandia Corporation.

Available to DOE and DOE contractors from
U.S. Department of Energy
Office of Scientific and Technical Information
P.O. Box 62
Oak Ridge, TN 37831

Telephone: (865) 576-8401
Facsimile: (865) 576-5728
E-Mail: reports@adonis.osti.gov
Online ordering: http://www.osti.gov/bridge

Available to the public from
U.S. Department of Commerce
National Technical Information Service
5285 Port Royal Rd.
Springfield, VA 22161

Telephone: (800) 553-6847
Facsimile: (703) 605-6900
E-Mail: orders@ntis.fedworld.gov
Online order: http://www.ntis.gov/help/ordermethods.asp?loc=7-4-0#online

SAND2010-5903

Unlimited Release

Printed May 2013

Trilinos Developers SQE Guide: ASC Software Quality Engineering Practices
Version 3.0

JAMES M. WILLENBRING, MICHAEL A. HEROUX
Sandia National Laboratories
P.O. Box 5800
Albuquerque, New Mexico 87185

ABSTRACT

The Trilinos Project is an effort to develop algorithms and enabling technologies within an object-oriented software framework for the solution of large-scale, complex multi-physics engineering and scientific problems. A new software capability is introduced into Trilinos as a *package*. A Trilinos package is an integral unit and, although there are exceptions such as utility packages, each package is typically developed by a small team of experts in a particular algorithms area such as algebraic preconditioners, nonlinear solvers, etc.

The Trilinos Developers SQE Guide is a resource for Trilinos package developers who are working under Advanced Simulation and Computing (ASC) and are therefore subject to the ASC Software Quality Engineering Practices as described in the Sandia National Laboratories Advanced Simulation and Computing (ASC) Software Quality Plan: ASC Software Quality Engineering Practices Version 3.0 document [1]. The Trilinos Developer Policies webpage [2] contains a lot of detailed information that is essential for all Trilinos developers. The Trilinos Software Lifecycle Model [3] defines the default lifecycle model for Trilinos packages and provides a context for many of the practices listed in this document.

ACKNOWLEDGEMENTS

The authors of Version 3.0 would like to recognize Robert Heaphy who was an author of the previous versions of this document. Previous versions of this document were called the *Trilinos Developers Guide Part II: ASC Software Quality Engineering Practices*. The title was changed for this version to recognize that the *Trilinos Developers Guide* has been retired and replaced with content on the Trilinos Developer Webpage.

1. Introduction

One objective of Advanced Simulation and Computing (ASC) is to develop professional software quality engineering (SQE) practices that will ensure the quality of software developed with ASC funding. To this end, the Sandia National Laboratories Advanced Simulation and Computing (ASC) Software Quality Plan: ASC Software Quality Engineering Practices Version 3.0 document [1], lists 30 practices that should be addressed by ASC software developers. The Trilinos Developers SQE Guide addresses each of the 30 practices, often referring to other Trilinos documentation except for issues that are unique to ASCI SQE practices. Although all Trilinos developers are encouraged to adopt these ASC practices, developers are only required to do so when working on ASC funded capabilities. Each of the 30 practices is discussed in terms of the responsibilities of the Trilinos Framework and each individual Trilinos Package. As can be seen from the table in Section 3, the Trilinos Framework provides a valuable service to the Trilinos Packages, providing package developers with ready-made support for many of the 30 practices, leaving to package developers only those practices that should be under the direction of each package team. In fact, eleven of the practices are the sole responsibility of the framework and the framework provides significant support for the remaining nineteen practices.

2. Roles, Documents, Tools and Events

The primary content of this document is a large table in Section 3, listing and discussing each of the 30 ASC SQE practices. Throughout the discussion, a number of roles (people), documents, tools and events are cited frequently. We list each of them here and assign acronyms where appropriate:

Roles:

- **ASC Program Management:** The ASC program (which includes ASC Algorithms) has an evolving set of processes for SQE. Many of the SQE processes in the Project Planning, Tracking, and Oversight, Risk Management, and Determination of Applicable Practices and level of Formality phases are driven by decisions made by ASC Program Management.
- **ASC Algorithms Program Element Lead and PI:** The Trilinos Project receives a significant portion of its funding from the Algorithms portion of the ASC Program. As a result, the ASC Algorithms Program Element Lead and PI play an important role in the Project Planning, Tracking, and Oversight, Risk Management, and Determination of Applicable Practices and level of Formality, following the guidelines and requirements established by ASC Program Management.
- **Trilinos Project Leader:** A fundamental management principal of the Trilinos Project is that packages should be as autonomous as possible, recognizing that local control with careful attention to interfaces is often the most effective approach to producing high quality software. At the same time there is a need for a single focal point in some situations. The Trilinos Project leader is the primary focal point for major project decisions. The duties of this person include:

1. Arbitrating in cases where a consensus decision cannot be reached as part of inter-package decisions.
2. Organizing Trilinos project events (see below).
3. Managing and tracking progress on the Trilinos Framework Responsibilities as described in Section 3.

- **Trilinos Release Manager:** The Trilinos Release Manager is responsible for tagging and branching the repository and generally coordinating the release process.
- **Trilinos Capability Leaders:** Because of the broad scope of Trilinos, we have leaders assigned to the following capability areas:
  1. Framework, Tools & Interfaces.
  2. Software Engineering Technologies and Integration
  3. I/O Support
  4. Discretizations.
  5. Meshes, Geometry, & Load Balancing.
  6. Scalable Linear Algebra.
  7. Linear & Eigen Solvers.
  8. Embedded Nonlinear Analysis Tools.

  These leaders have responsibilities across packages and are responsible for strategic planning for Trilinos in their areas.
- **Package Leader:** Each Trilinos package has one or more leaders. These leaders are responsible for managing and tracking package responsibilities as described in Section 3. These leaders are also responsible for attending the Monthly Trilinos Leaders Meetings, representing the package development team and disseminating information to the team. Only leaders of packages funded by ASC are responsible for all of the practices listed in Section 3.
- **Package Developer:** Each package has an identifiable group of developers. Any given individual may be a member of multiple package development teams.
- **Framework Developer:** The Trilinos Framework also has an identifiable group of developers. These individuals may also be members of package development teams.

Documentation:

- **Trilinos Developer Website (TDW):** Primary development documentation for Trilinos developers. Discusses policies that apply to all Trilinos developers and describes the services available to packages that are part of Trilinos [2].
- **Trilinos Software Lifecycle Model:** Defines a 3-phase promotional lifecycle model that recognizes the changing requirements for Trilinos capabilities as work goes from proof-of-concept to production quality. The three phases are research, production growth, and production maintenance. To transition from one phase to the next, a promotional event occurs. Most often, packages transition from phase to phase as a whole, but the model also allows for different capabilities within packages to be in different phases, provided certain conditions are met [3].
- **Trilinos Strategic Plan (TSP):** Lists the strategic goals of the project and provides pointers to other important information, such as a list of ASC stakeholders and project capabilities [4].

- **Trilinos Project Plan (TPP):** Each fiscal year the ASC Algorithms team gathers user and software requirements from ASC application teams and its own members. The exact form and number of documents generated has changed from year to year, as the ASC program itself defines its processes for software quality. Currently the TPP is called the *ASC Algorithms Inner Core (AIC) Project Plan* [5]. In past years we have provided all analysis and documentation request by ASC program managers, and have developed additional documents that provide greater detail than what ASC requires. As ASC program practices evolve, we will continue to adapt our documents and processes to match, especially in the Requirements, Project Planning, Tracking and Oversight and Risk Management phases. Although the type and number of documents required by the ASC program has and will continue to change, for simplicity we refer to entire collection of these documents and related documents that provide greater detail as the Trilinos Project Plan. In the recent past, the primary documents have been the Objectives and Resource plan for ASC Algorithms R&D, the ASC Algorithms Implementation Plan and the ASC Algorithms Objectives and Resource Plan.
- **ASC Algorithms Quarterly Report (AAQR):** At the end of each quarter, when requested by ASC program management, the ASC Algorithms team will generate a progress report and list adjusted milestones as needed.
- **Package Specific Documentation (PSD):** In the Package Responsibilities column we discuss in detail how a package should satisfy a particular practice. However, it is always the case that a package may satisfy any given practice via its process as long as the alternate process is documented in the appropriate Package Specific Document. For example, if Package X adopts a lifecycle that is different than The Trilinos Software Lifecycle Model, the alternative lifecycle model should be documented.

Key point: It is always the case that a package may satisfy any given practice via its own process as long as the alternate process is documented in the appropriate Package Specific Document.

Tools:

- **Git (Fast Version Control System)**: The Trilinos source code repository is maintained using git [6]. The git repository resides on the Trilinos Development platform "software.sandia.gov" on the Sandia Open Network (SON). Sensitive documents are retained under a separate repository on the Sandia Restricted Network (SRN), as necessary. Use of git is documented on the TDW.
- **Concurrent Versions Systems (CVS):** Source code for the Trilinos website, Trilinos SQE documents, and other Trilinos-related repositories and documents are maintained using CVS [7]. The CVS repository resides on the Trilinos Development platform "software.sandia.gov" on the Sandia Open Network (SON). Use of CVS is documented on the TDW.
- **Bugzilla:** All major features and software faults are reported using Bugzilla [8], a web-based issue-tracking package. Each Trilinos package, the Trilinos framework and Bugzilla itself are set up as Bugzilla products. Bugzilla is available at

http://software.sandia.gov/bugzilla. Use of Bugzilla is documented on the TDW. Note that the processes for reporting bugs, tracking issues, and requesting enhancements are built directly into Bugzilla.

- **Mailman:** Each Trilinos package has a set of Mailman [9] mail lists to support communication and archiving of important information and artifacts. List descriptions are documented on the TDW.
- **Doxygen:** Many Trilinos packages use the documentation-generating package called Doxygen [10]. Doxygen processes source code comments producing detailed online and printed documentation of the processed source. Although it can be used with many languages and is highly configurable, the most common use of it within Trilinos is to process header files containing description of C++ classes and detailed documentation of the major user-callable methods in each class. Doxygen also extracts information about class interactions and dependencies. Most Trilinos packages presently use Doxygen to provide user reference documentation.
- **CMake/CTest:** Trilinos configuration and building is facilitated by using CMake [11]. CTest is a tool used to invoke and organize the Trilinos test suite. CMake facilitates dynamic configuration and building of Trilinos across a broad set of computer platforms. Via configure-time compilation and linking tests, queries to the operating system and user-specified parameters, Trilinos can be configured and built on almost any platform with minimal user knowledge of details such as location of system libraries and compilers. The CMake build system also supports installation of Trilinos for multiple users and automatic creation of distribution tar files and installers.
- **CDash:** Trilinos testing results (including nightly, continuous integration, experimental, etc.), reported by CTest, are stored and displayed using CDash [12]. Information about the test results webpage is available on the TDW.
- **software.sandia.gov:** The primary Trilinos development platform is a Linux server called software.sandia.gov. This platform supports many of the tools listed above, contains the Trilinos git and CVS repositories and all of the Mailman and Bugzilla archives. This platform is on the Sandia Open Network (SON), and is accessible from any internet-connected machine. The Sandia CSUSP Unclassified backup "Legato" system is used for backing up software.sandia.gov.
- **trilinos.sandia.gov:** The primary user-oriented website, containing most user documentation and download instructions. The machine that hosts trilinos.sandia.gov also hosts the CDash Dashboard and the database used to track downloads.
- **Trilinos Process Checklists:** The Trilinos project team has a number of processes defined via process checklists. A list of current and past process checklists is available on the TDW [13]. Completed checklists related to releases and CVS commits are stored in Bugzilla, either in the body of the bug, or as an attachment. Once a process is started, it can be completed using the version of the process checklist that was current at the time the process was initiated, unless otherwise requested by the Trilinos Project Leader.

Events:

- **Monthly Trilinos Leaders Meeting:** Package leaders for Trilinos packages, Trilinos management and other stakeholders are invited to participate in a monthly leadership meeting. Meeting minutes are sent to the Trilinos-Leaders@software.sandia.gov mail list

for communication and archiving. Meeting topics include discussion of Trilinos policies, requirements, design, implementation, testing and documentation. We also conduct developer training as needed during these meetings. A request for agenda items and a meeting agenda is sent to the Trilinos-Leaders mail list prior to each meeting.

- **Annual Trilinos User Group (TUG) Meeting:** Approximately once a year we hold a meeting for Trilinos users. During this meeting we present an overview of Trilinos, and detailed presentations of Trilinos packages. The last day of the meeting is typically reserved for Trilinos Developer-oriented presentations and discussions. The seventh annual meeting was held in November, 2009.
- **Annual Trilinos Spring Developer Meeting:** Approximately once a year we hold a meeting for Trilinos Developers. Typical topics include training and other presentations as well as strategic planning. The second annual meeting was held in May, 2010.
- **Quarterly Trilinos Advisors Meeting:** A small group of Trilinos stakeholders participate in a quarterly Trilinos Advisory Group meeting. These meetings give Trilinos developers the chance to discuss important issues with key users, and give users the opportunity to directly share concerns and suggestions with developers. Users are asked to commit to the group for a period of just over one year, spanning from before TUG, to just after TUG the following year. (The number of non-developer participants in the group is doubled for TUG so the outgoing group can interface with the incoming group.) Meeting minutes are sent to the Trilinos-Advisors@software.sandia.gov mail list for communication and archiving. A request for agenda items and a meeting agenda is sent to the Trilinos-Advisors mail list prior to each meeting.
- **Quarterly Trilinos Capability Leaders Meeting:** The Trilinos Capability Leaders meet quarterly to discuss high level Trilinos issues including the general direction of the project. Meeting minutes are sent to the Trilinos-Board@software.sandia.gov mail list for communication and archiving. A request for agenda items and a meeting agenda is sent to the Trilinos-Board mail list prior to each meeting.

3. Trilinos Practices Table

The remainder of this document is a table that follows, item by item, the 30 practices listed in the ASC Software Quality Engineering Practices document. For each practice, the responsibilities of the Trilinos Framework and each Trilinos Package are described using present-tense phrasing. Please note that some of these responsibilities are not fully addressed at this time, in which case this document serves as a plan rather than a statement of practice.

Key point: … some of these responsibilities are not fully addressed at this time, in which case this document serves as a plan rather than a statement of practice.

Practice	Trilinos Framework Responsibilities	Trilinos Package Responsibilities
Project Management		
1. Strategic Planning		
PR1. Document and maintain a strategic plan.	The Trilinos Strategic Plan [4] is maintained at the Trilinos project level with input from packages.	Provide input for the Trilinos Strategic Plan as appropriate.
2. Determination of Applicable Practices and Level of Formality		
PR2. Perform a risk-based assessment, determine level of formality and applicable practices, and obtain approvals.	Trilinos project level risk identification and mitigation is included in the TPP and AAQR, when and as requested by ASC program management. An ASC risk-based assessment has been completed and approved.	For packages in the Research phase the associated technical risk cannot be mitigated by a high level of formality, so by default these packages follow a low level of formality (LOF). During the promotional events defined by The Trilinos Software Lifecycle Model, risk identification is required. By default, the LOF at the Production Growth phase is medium. At the Production Maintenance phase, the default LOF is high. The LOF may be modified from the default level based on risk assessments at the package level (see lifecycle model promotional events).
3. Process Implementation and Improvement		
PR3. Document lifecycle processes and their interdependencies, and obtain approvals.	The Trilinos Software Lifecycle Model defines the lifecycle for a Trilinos package.	None.

Practice	Trilinos Framework Responsibilities	Trilinos Package Responsibilities
PR4. Define, collect, and monitor appropriate process metrics.	Trilinos process checklists [13] require and recommend the collection of a number of project metrics. Other metrics, such as build and test failures and coverage rates are collected as part of standard Trilinos testing using CTest and CDash. Certain metrics, such as build failures, are monitored frequently. There is an ongoing effort to define new metrics for the project.	Package teams monitor metrics collected at the Trilinos level, such as coverage statistics and build failures and collect and monitor metrics of specific package interest.
PR5. Periodically evaluate quality problems and implement process improvements.	Many process checklists use Plan, Do, Check, Act, which provides built in process improvement. Other checklists are reviewed periodically and improved upon. Process improvements are discussed at TUG and at the Trilinos Spring Developer Meeting.	Package level process checklists utilize process improvement in the same way as Framework level process checklists. Metrics are made available to package development teams to allow them to make package level process improvements.
4. Requirements Engineering		
PR6. Identify stakeholders and other requirements sources.	The TSP and TPP identify Trilinos stakeholders.	None.
PR7. Gather and manage stakeholders' expectations and requirements.	Primary user requirements are gathered and documented in the TPP after negotiation by level 1 management. In addition, topics discussed during the monthly Trilinos Leader's meeting include Trilinos user requirements. Meeting minutes are archived on the trilinos-leaders@software.sandia.gov mail list. Select platter deliverables are defined from higher-level requirements and receive special tracking by management.	None.
PR8. Derive, negotiate, manage, and trace requirements.	The TPP covers the derivation, negotiation, management, and tracing of requirements. The AAQR documents the managing and tracing of and the success in meeting requirements.	None.
5. Risk Management		
PR9. Identify and analyze risk events.	Risk identification and analysis is included in the TPP and AAQR, when and as requested by ASC program management.	During the promotional events defined by The Trilinos Software Lifecycle Model, risk identification is required. Include identified risks in quarterly update to ASC Algorithms PI.
PR10. Define, monitor, and implement the risk response.	Risk mitigation and response is included in the TPP and AAQR, when and as requested by ASC program management. Identified risks are considered when implementing new policies and tools.	Provide quarterly updates to the ASC Algorithms PI.

Practice	Trilinos Framework Responsibilities	Trilinos Package Responsibilities
6. Project Planning, Tracking and Oversight		
PR11. Create and manage the project plan.	The TPP is maintained at the project level.	None.
PR12. Track performance versus project plan and implement needed (corrective) actions.	The AAQR is written and submitted quarterly, when requested by ASC program management.	None.
Software Engineering		
7. Software Development		
PR13. Communicate and review design.	Doxygen [10], user mail lists and developer mail lists are provided on the Trilinos development platform software.sandia.gov. Trilinos Framework level design discussions occur at Trilinos Framework Developer meetings and on the trilinos-framework mail list. All mail list traffic is subject to peer review. External reviews occur naturally because of the open source Trilinos release.	This practice is addressed by the lifecycle model. Briefly, design at research phase is typically captured in a notebook or on a mail list. During the production growth phase, the design is often communicated and reviewed during face to face or phone meetings (and minutes are sent to a mail list), or on a developer mail list. During the production maintenance phase, formal documentation and review of design will occur. In all phases design is also captured in the form of Doxygen documentation. In most cases, the Doxygen documentation represents a true design as it is produced before the associated code is written.
PR14. Create required software and product documentation.	A CVS repository is provided for all code and artifacts. Doxygen [10] is provided on the Trilinos development platform software.sandia.gov. Trilinos level documentation is provided at the Trilinos website.	Packages follow the adopted lifecycle for software development.
8. Integration of Third Party or Other Software		

Practice	Trilinos Framework Responsibilities	Trilinos Package Responsibilities
PR15. Identify and track third party software products and follow applicable agreements.	Supported versions of third-party software can be kept in the Trilinos3PL CVS repository. Third-party software that is widely available and adheres to established standards such as BLAS, LAPACK, and MPI are not kept under version control. Trilinos supports a wide array of versions and vendors of these common libraries that adhere to the appropriate standard. A list of third-party software used by Trilinos is kept in Trilinos/cmake/TrilinosTPLs.cmake. This list is processed by the CMake build system when looking for TPLs and associated options at configure time.	Check new supported versions of 3PL's into the Trilinos3PL repository and/or document supported versions. Follow all applicable license agreements.
PR16. Identify, accept ownership, and manage assimilation of other software products.	Trilinos packages do not generally accept ownership of third-party software. If a package team chooses to do so, they have the option to use the Trilinos or Trilinos3PL repository to store the code.	Follow all applicable license agreements. Assimilated code should be maintained like core package code.
9. Configuration Management		
PR17. Perform version control of identified software product artifacts.	A git repository is maintained for all Trilinos packages. Package-checkins mail lists archive all product modifications. Trilinos release versioning is handled by release and release update process checklists [13]. A CVS repository is maintained for the website, SQE documents, etc.	Package developers utilize the Trilinos git and CVS repositories.
PR18. Record and track issues associated with the software product.	A Bugzilla product is provided for each Trilinos package. All issues are tracked via Bugzilla and the underlying MySQL [14] database. Provide documentation for using Bugzilla. For the Trilinos Framework, a product backlog is currently maintained within Bugzilla.	Package developers, or their customers, file issue reports using the Trilinos Bugzilla site. This includes major feature requests and software problems. Issue reports are kept up-to-date.
PR19. Ensure backup and disaster recovery of software product artifacts.	The Trilinos development platform, software.sandia.gov is backed up regularly by the Sandia CSUSP Unclassified backup "Legato" system. All git, CVS, Mailman, and Bugzilla data and artifacts are retained indefinitely. Each clone of a git repository is a complete repository. System administrators perform a test recovery from a Legato backup at least yearly.	None.
10. Release and Distribution Management		

Practice	Trilinos Framework Responsibilities	Trilinos Package Responsibilities
PR20. Plan and generate the release.	Release requests are negotiated between the Trilinos Project Leader and customers. A release cycle commences when an announcement of a release target date is sent to the Trilinos-developers mail list. This email describes the release plans, or the plan is discussed as part of the monthly Trilinos Leaders Meeting. Part of the Trilinos Level Release Process Checklist and Trilinos Web Release Process Checklist address release generation.	Part of the Trilinos Package Level Release Process Checklist addresses release generation. All packages that are included in an external release for the first time are required to complete this checklist. Package teams indicate permission to release their packages for releases after the initial external release.
PR21. Certify that the software product (code and its related artifacts) is ready for release and distribution.	Part of the Trilinos Level Release Process Checklist and Trilinos Web Release Process Checklist address release certification. Part of the required certification is provided by major customers.	Part of the Trilinos Package Level Release Process Checklist addresses release certification. Package leaders certify that their package is ready for release by providing permission to release.
PR22. Distribute release to customers.	Part of the Trilinos Level Release Process Checklist, Release Update Checklist, and Trilinos Web Release Process Checklist address release distribution.	None.
11. Customer Support		
PR23. Define and implement a customer support plan.	Customer support is addressed in the TPP, and the TSP. Frequent communication, the Trilinos User Group meeting, tutorials at conferences, the Trilinos website, Trilinos Tutorial, any existing package user guides, Bugzilla, the lifecycle level of formality, the trilinos-help and trilinos-user mail lists, and the list of developer contacts available online are all important components of Trilinos customer support.	Responsible for providing customer support at the package level. To propose a new Trilinos package, an email is sent to the trilinos-developer list. Topics covered in the email include who will maintain the package and long-term package support.
PR24. Implement the training identified in the customer support plan.	Organize the annual Trilinos User Group meeting. Support developers offering tutorials at conferences. Maintain the Trilinos website and Trilinos Tutorial (document and online version).	Create and maintain package documentation. Provide tutorials at the Trilinos User Group meeting as appropriate.
PR25. Evaluate customer feedback to determine customer satisfaction.	Users provide feedback at Trilinos Advisory Group meetings and via email, issue reports, and conversations throughout the year. They are also invited to TUG where users are given the chance to provide feedback. In addition, at least one user is invited to give a presentation at TUG that often includes comments on their level of satisfaction and opportunities for improvement. Occasionally, customer surveys are conducted.	Respond promptly to user issues. Package specific surveys can be conducted, when appropriate.

Practice	Trilinos Framework Responsibilities	Trilinos Package Responsibilities
Software Verification		
12. Software Verification		
PR26. Develop and maintain a software verification plan.	The Trilinos software verification plan is included in the TPP. The Trilinos website includes documentation for and a discussion of the Trilinos Test Harness to assist developers in setting up package testing, including a Test harness page (https://software.sandia.gov/trilinos/developer/test_harness/index.html) and testing policies accessible from https://software.sandia.gov/trilinos/developer/policies/index.html.	None.
PR27. Conduct tests to demonstrate that acceptance criteria are met and to ensure that previously tested capabilities continue to perform as expected.	The results from nightly test cases are automatically mailed to the appropriate package-regression@software.sandia.gov mail list. The results are archived. The completion of acceptance tests that are run by users before a release are also archived via the trilinos-framework@software.sandia.gov and/or the trilinos-developers@software.sandia.gov mail list. Release testing is addressed by part of the Trilinos Level Release Process Checklist. Multiple customers run nightly tests against the Trilinos development branch and promptly report any failures or regressions. A script is maintained by the framework that provides pre-checkin testing for developers. After repository changes, a continuous integration build is run to report any failures or regressions before the next round of nightly testing.	Package developers are responsible for deciding what tests need to be written for their packages. Part of the Trilinos Package Level Release Process Checklist addresses testing that must be completed prior to the initial external release. Developers resolve issues that are found with high priority.
PR28. Conduct independent technical reviews to evaluate adequacy with respect to requirements.	Framework list discussions and peer reviews occur as appropriate. Feedback on the adequacy of existing and new framework components is gathered at TUG and Trilinos Monthly Leaders meetings.	The formality of technical reviews depends on which phase of the lifecycle model package is in. During the research phase, this often means publishing results. During the production maintenance phase, formal inspections occur as appropriate. During the production growth phase publications are produced and/or inspections occur as appropriate.
Training		
13. Training		

Practice	Trilinos Framework Responsibilities	Trilinos Package Responsibilities
PR29. Determine project team training needed to fulfill assigned roles and responsibilities.	Training, based on the needs determined by the Trilinos Project Leader or developer interest, is provided during developer day at the Trilinos User Group meeting and at the Trilinos Spring Developer meeting. Training is also performed as needed during the monthly Trilinos Leaders Meeting and during special events, such as the Software Engineering Seminar Series. Training events are announced on the trilinos-leaders or trilinos-developers mail list prior to the training session. A checklist is provided for new developers and is to be completed with input from an existing Trilinos developer.	None.
PR30. Track training undertaken by project team.	Training is tracked by checking attendance lists into the TrilinosSQE CVS repository, or via meeting minutes sent to the appropriate mail list.	None.

References

[1] Molly Minana, Jennifer Turgeon, Martin Pilch, and Patricia Hackney, *Sandia National Laboratories Advanced Simulation and Computing (ASC) Software Quality Plan: ASC Software Quality Engineering Practices, Version 3.0*, Sandia National Laboratories, SAND2008-5517, January 2009.

[2] Trilinos Developer Website: https://software.sandia.gov/trilinos/developer.

[3] J. Willenbring, R. Heaphy, M. Heroux and M. Phenow, *The Trilinos Software Lifecycle Model*, Sandia National Laboratories, SAND2006-6929, November 2006.

[4] M. Heroux, J. Willenbring and R. Heaphy, *Trilinos Project Strategic Plan*, Sandia National Laboratories, SAND2008-0479, Updated June 2010.

[5] K. Alvin, *ASC Algorithms Inner Core (AIC) Project Plan, Revision 2*, Sandia National Laboratories, September 2009.

[6] Git Home Page: http://git-scm.com/.

[7] Gnu CVS Home Page: http://www.gnu.org/software/cvs.

[8] Mozilla Bugzilla Home Page: http://www.mozilla.org/projects/bugzilla.

[9] Mailman Home Page: http://www.gnu.org/software/mailman.

[10] Doxygen Home Page: http://www.doxygen.org.

[11] CMake Home Page: http://www.cmake.org.

[12] CDash Home Page: http://www.cdash.org.

[13] Trilinos Process Checklists: http://software.sandia.gov/trilinos/developer/sqp/checklists/index.html.

[14] MySQL Home Page: http://www.mysql.com.

DISTRIBUTION

1	MS0899	Technical Library	9536 (electronic copy)


Sandia National Laboratories