Imperial College London

Nektar++
Community and Development Strategy

Chris Cantwell, David Moxey, Mike Kirby, Spencer Sherwin

Nektar++ Workshop
Imperial College London
7th June 2016

NEKTAR++
SPECTRAL/HP ELEMENT FRAMEWORK

Overview

- Nektar++ Community and Support
  - What's new?
  - Accessibility
  - Education
  - Support
- Improving code stability
  - Developer practices
  - Release schedule
  - Review process
- Summary and What's next?

Nektar++ Community and Support

The mission of Nektar++ is to make spectral/hp element methods accessible to a broader audience and remove the implementation complexity barrier which limits adoption of these methods.

We are keen to engage with the community of users in their use of Nektar++.

Accessibility
(support for range of OSs, packages)

Education
(tutorials, documentation)

Support
(mailing list, building a community)

What's New?

- **Tutorials** for new users (more later)
- Redevelopment of **Nektar++ website** to be more community driven
- User guide available online in HTML form
- **Binary packages** now available for a broader range of platforms
- **Tested** on a wider and more up-to-date range of operating systems

- Now used on a range of **supercomputers**:
  - SGI ICE (CX2, Imperial College London)
  - Blue Gene/Q (Mira, Argonne National Laboratory)
  - Cray XC30 (ARCHER, UK National Supercomputer)
  - Cray XE6 (Copper, DoD)

Accessibility

How can one get Nektar++?

- Binary packages for Linux and OSX




Accessibility

How can one get Nektar++?

- Binary packages for Linux and OSX

- Homebrew and MacPorts (OSX):
  - See `http://www.nektar.info/installing-nektar-on-os-x-through-macports-or-homebrew/`

Accessibility

How can one get Nektar++?

- Binary packages for Linux and OSX

- Homebrew and MacPorts (OSX):
  - See `http://www.nektar.info/installing-nektar-on-os-x-through-macports-or-homebrew/`
- Source code bundles (self-compile) from `http://www.nektar.info`
  - Different archive formats (.tar.gz, .tar.bz2, .tar.xz, .zip)
  - Regular releases, but not most recent updates
  - Build against Nektar++ libraries, but no direct means to contribute

Accessibility

How can one get Nektar++?

- Binary packages for Linux and OSX

- Homebrew and MacPorts (OSX):
  - See `http://www.nektar.info/installing-nektar-on-os-x-through-macports-or-homebrew/`
- Source code bundles (self-compile) from `http://www.nektar.info`
  - Different archive formats (.tar.gz, .tar.bz2, .tar.xz, .zip)
  - Regular releases, but not most recent updates
  - Build against Nektar++ libraries, but no direct means to contribute
- Git repository
  - Anonymous checkout

    ```
    git clone http://gitlab.nektar.info/clone/nektar/nektar.git
    ```
  - Authenticated checkout

    ```
    git clone git@gitlab.nektar.info:nektar/nektar.git
    ```

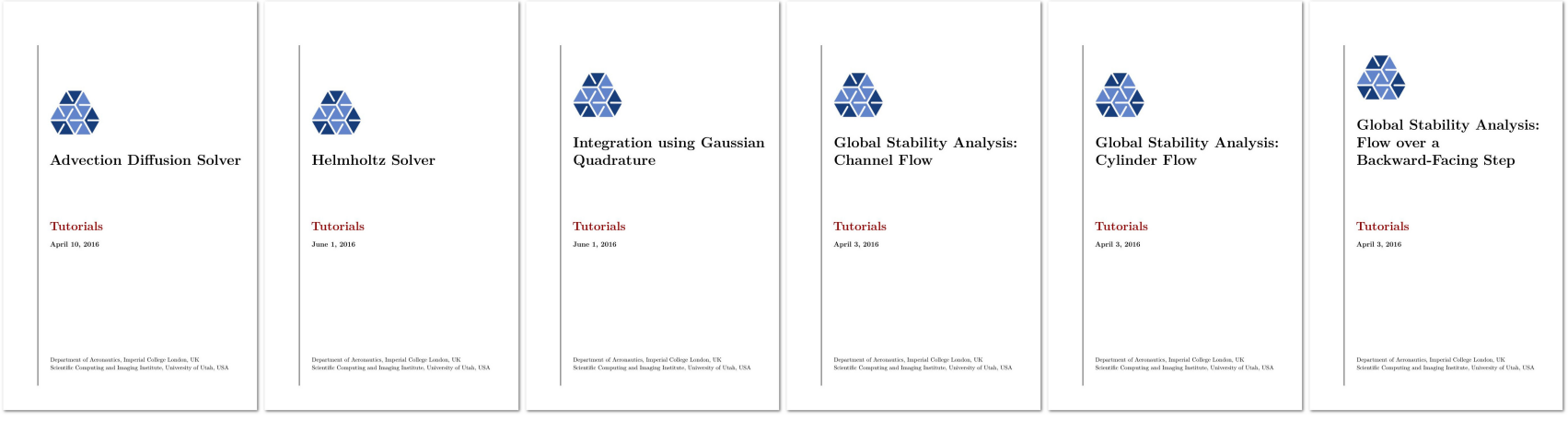
Education: Tutorials

- Intended to help new users quickly get started with Nektar++
- Low barrier to entry
- Introduce various aspects of the code
  - Basic usage of the solvers
  - Use of the code for specific applications
  - Using the libraries to support writing new code

Education: Tutorials

- Intended to help new users quickly get started with Nektar++
- Low barrier to entry
- Introduce various aspects of the code
  - Basic usage of the solvers
  - Use of the code for specific applications
  - Using the libraries to support writing new code
- Available both as PDF documents and HTML webpages

  `http://www.nektar.info/community/tutorials/`
- Current tutorial selection

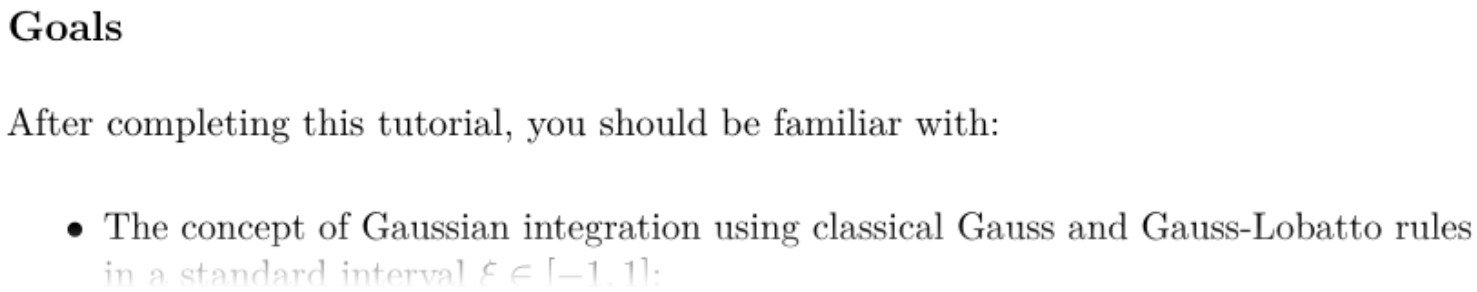


- We welcome suggestions for new tutorials!

Education: Tutorials

Each tutorial provides a structured introduction to a topic:

- Introduction and goals



Goals

After completing this tutorial, you should be familiar with:

- The concept of Gaussian integration using classical Gauss and Gauss-Lobatto rules in a standard interval $\xi \in [-1, 1]$:

Education: Tutorials

Each tutorial provides a structured introduction to a topic:

- Introduction and goals

Goals

After completing this tutorial, you should be familiar with:

- The concept of Gaussian integration using classical Gauss and Gauss-Lobatto rules in a standard interval $\xi \in [-1, 1]$:

- Context

the use of numerical integration or *quadrature*. The fundamental building block is the approximation of the integral by a finite summation of the form

$$\int_{-1}^{1} u(\xi) d\xi \approx \sum_{i=0}^{q-1} w_i u(\xi_i),$$

where w_i are specified constants or *weights* and ξ_i represents an abscissa of q distinct

Education: Tutorials

Each tutorial provides a structured introduction to a topic:

- Introduction and goals

Goals

After completing this tutorial, you should be familiar with:

- The concept of Gaussian integration using classical Gauss and Gauss-Lobatto rules in a standard interval $\xi \in [-1, 1]$:

- Context

the use of numerical integration or *quadrature*. The fundamental building block is the approximation of the integral by a finite summation of the form

$$\int_{-1}^{1} u(\xi) d\xi \approx \sum_{i=0}^{q-1} w_i u(\xi_i),$$

where w_i are specified constants or *weights* and ξ_i represents an abscissa of q distinct

- Step-by-step tasks, often with partially completed files

Task 3.3

Evaluate the previous integral for a quadrature order of $Q = Q_{\max}$ where $Q_{\max} = 7$ and 8 to verify that to exactly integrate with Gauss-Lobatto type integration you require an additional quadrature point and weights.

Education: Tutorials

Each tutorial provides a structured introduction to a topic:

- Introduction and goals

Goals

After completing this tutorial, you should be familiar with:

- The concept of Gaussian integration using classical Gauss and Gauss-Lobatto rules in a standard interval $\xi \in [-1, 1]$:

- Context

the use of numerical integration or *quadrature*. The fundamental building block is the approximation of the integral by a finite summation of the form

$$\int_{-1}^{1} u(\xi) d\xi \approx \sum_{i=0}^{q-1} w_i u(\xi_i),$$

where w_i are specified constants or *weights* and ξ_i represents an abscissa of q distinct

- Step-by-step tasks, often with partially completed files

Task 3.3

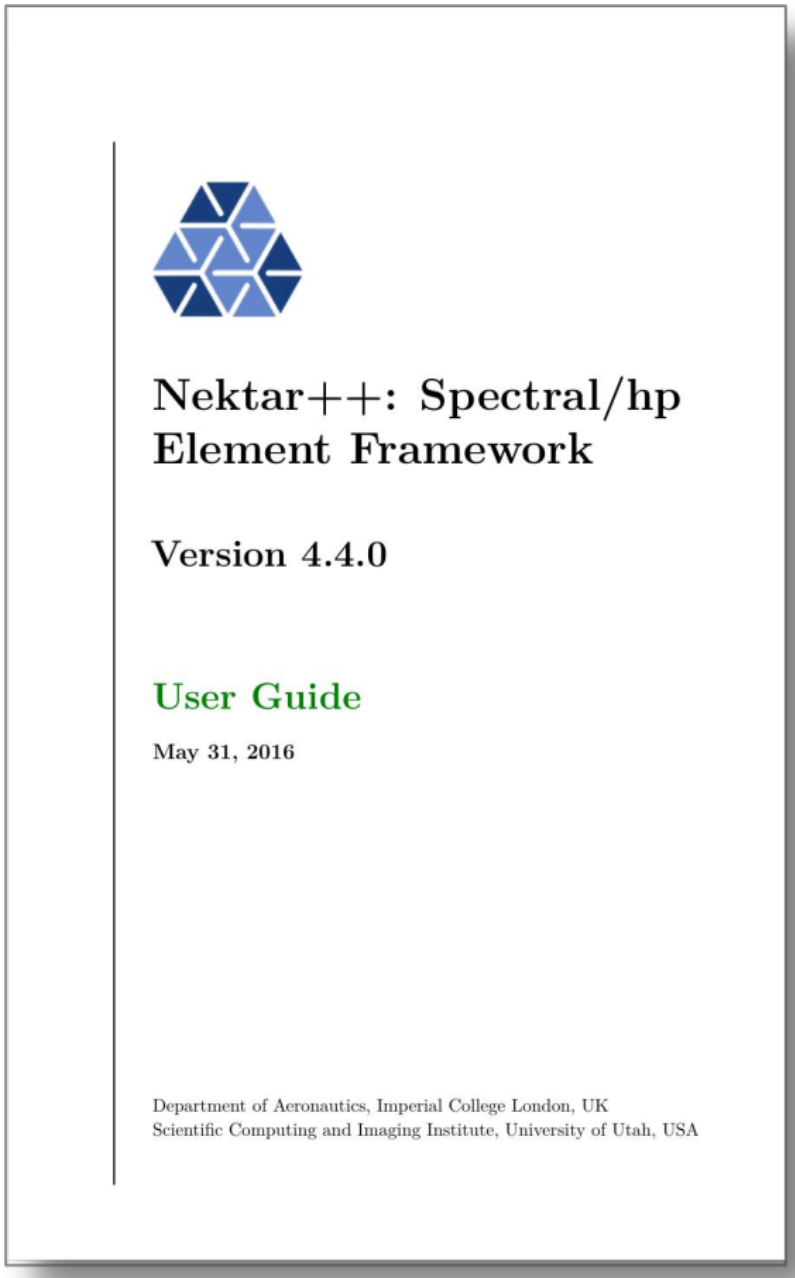
Evaluate the previous integral for a quadrature order of $Q = Q_{\max}$ where $Q_{\max} = 7$ and 8 to verify that to exactly integrate with Gauss-Lobatto type integration you require an additional quadrature point and weights.

- Summary of what has been learnt
- Set of complete solutions

Education: Documentation

- Versioned with code

Education: Documentation



Nektar++: Spectral/hp Element Framework

Version 4.4.0

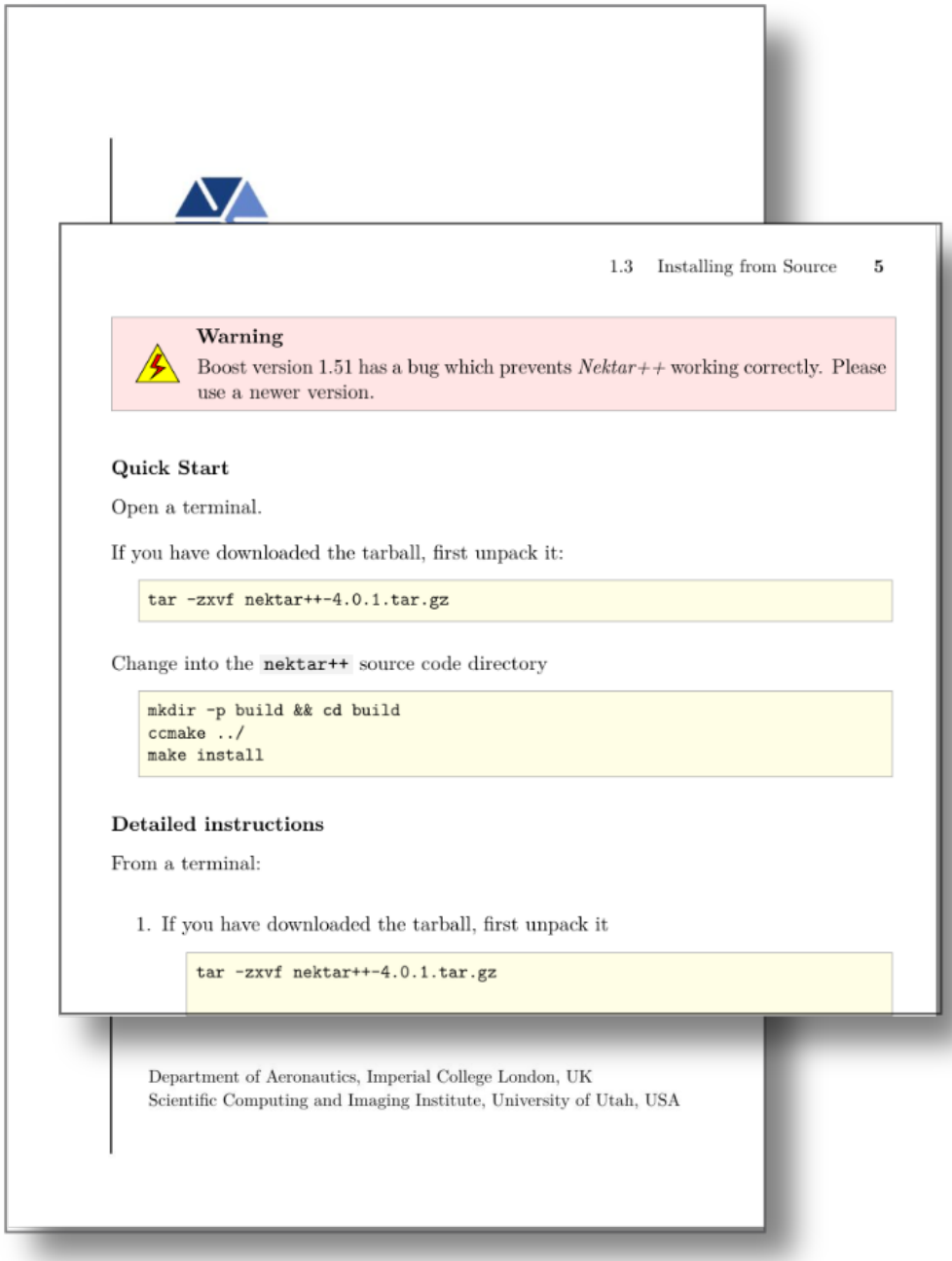
User Guide

May 31, 2016

Department of Aeronautics, Imperial College London, UK
Scientific Computing and Imaging Institute, University of Utah, USA

- Versioned with code
- User guide

Education: Documentation



1.3 Installing from Source 5

Warning
Boost version 1.51 has a bug which prevents *Nektar++* working correctly. Please use a newer version.

Quick Start

Open a terminal.

If you have downloaded the tarball, first unpack it:

```
tar -zxvf nektar++-4.0.1.tar.gz
```

Change into the `nektar++` source code directory

```
mkdir -p build && cd build
ccmake ../
make install
```

Detailed instructions

From a terminal:

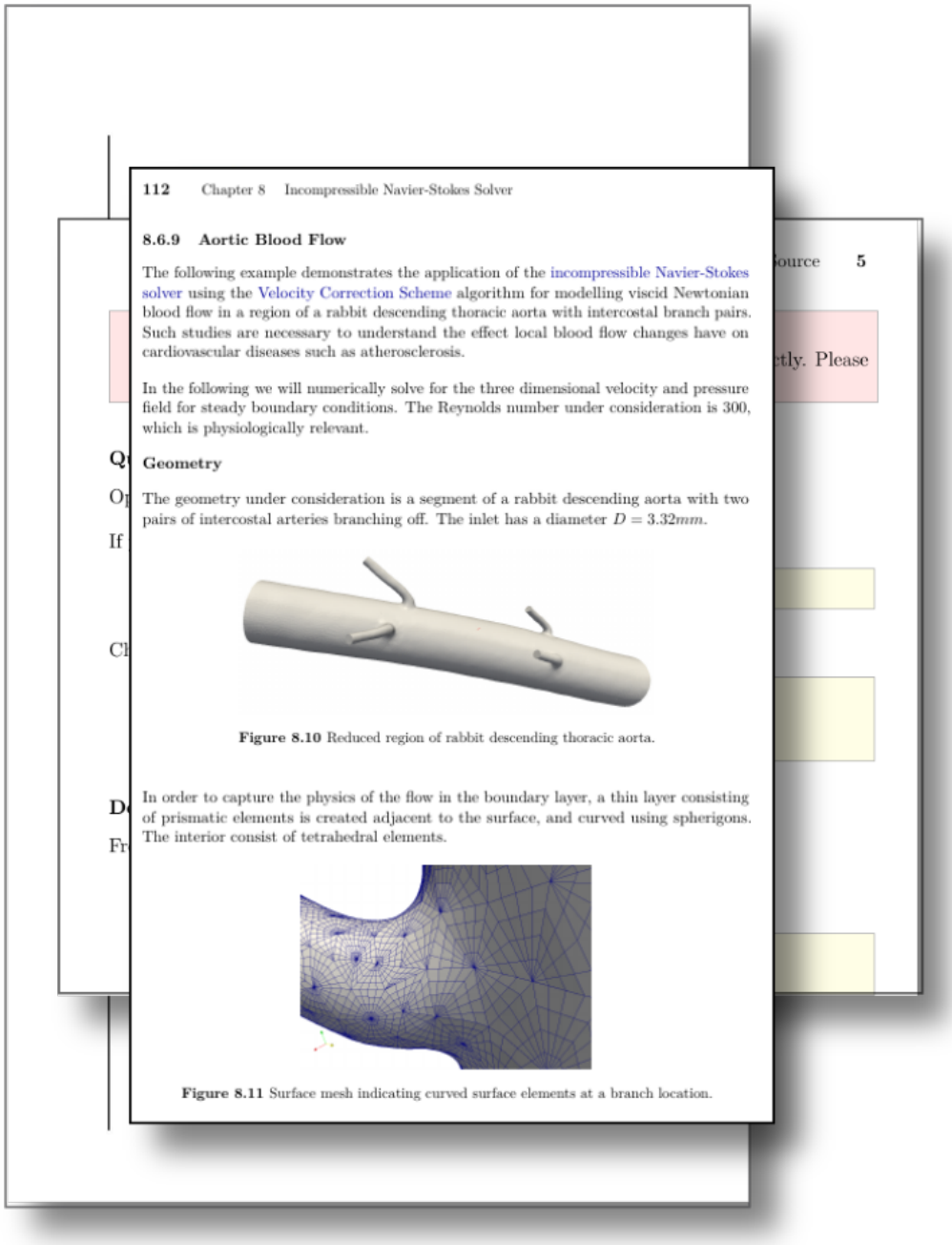
1. If you have downloaded the tarball, first unpack it

```
tar -zxvf nektar++-4.0.1.tar.gz
```

Department of Aeronautics, Imperial College London, UK
Scientific Computing and Imaging Institute, University of Utah, USA

- Versioned with code
- User guide
  - Installation

Education: Documentation



112 Chapter 8 Incompressible Navier-Stokes Solver

8.6.9 Aortic Blood Flow

The following example demonstrates the application of the incompressible Navier-Stokes solver using the Velocity Correction Scheme algorithm for modelling viscid Newtonian blood flow in a region of a rabbit descending thoracic aorta with intercostal branch pairs. Such studies are necessary to understand the effect local blood flow changes have on cardiovascular diseases such as atherosclerosis.

In the following we will numerically solve for the three dimensional velocity and pressure field for steady boundary conditions. The Reynolds number under consideration is 300, which is physiologically relevant.

Geometry

The geometry under consideration is a segment of a rabbit descending aorta with two pairs of intercostal arteries branching off. The inlet has a diameter $D = 3.32mm$.

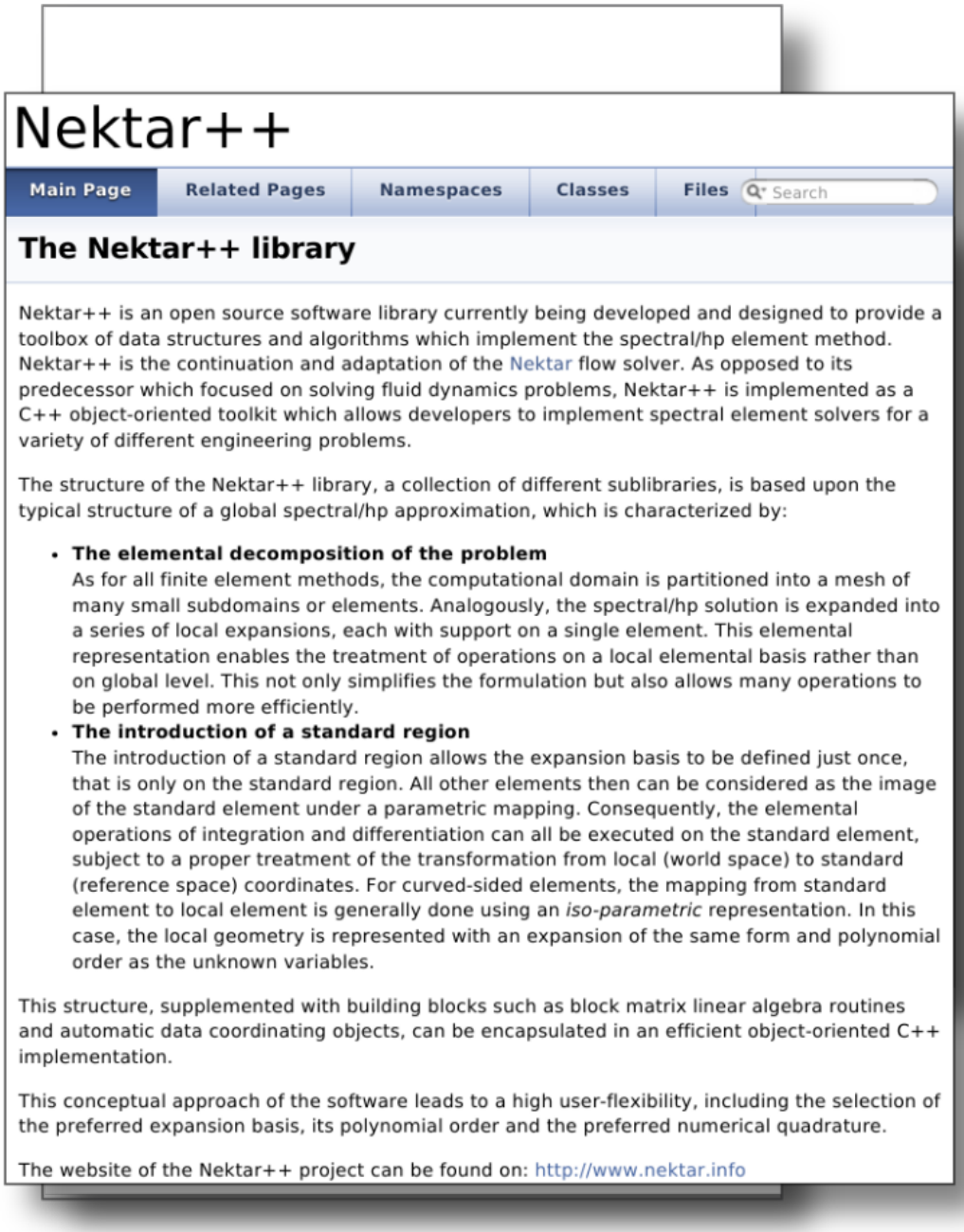
Figure 8.10 Reduced region of rabbit descending thoracic aorta.

In order to capture the physics of the flow in the boundary layer, a thin layer consisting of prismatic elements is created adjacent to the surface, and curved using spherigons. The interior consist of tetrahedral elements.

Figure 8.11 Surface mesh indicating curved surface elements at a branch location.

- Versioned with code
- User guide
  - Installation
  - Solver and Utility usage
  - Examples
  - Input file reference
  - FAQs

Education: Documentation



Nektar++

Main Page | Related Pages | Namespaces | Classes | Files | Search

The Nektar++ library

Nektar++ is an open source software library currently being developed and designed to provide a toolbox of data structures and algorithms which implement the spectral/hp element method. Nektar++ is the continuation and adaptation of the Nektar flow solver. As opposed to its predecessor which focused on solving fluid dynamics problems, Nektar++ is implemented as a C++ object-oriented toolkit which allows developers to implement spectral element solvers for a variety of different engineering problems.

The structure of the Nektar++ library, a collection of different sublibraries, is based upon the typical structure of a global spectral/hp approximation, which is characterized by:

- **The elemental decomposition of the problem**
  As for all finite element methods, the computational domain is partitioned into a mesh of many small subdomains or elements. Analogously, the spectral/hp solution is expanded into a series of local expansions, each with support on a single element. This elemental representation enables the treatment of operations on a local elemental basis rather than on global level. This not only simplifies the formulation but also allows many operations to be performed more efficiently.
- **The introduction of a standard region**
  The introduction of a standard region allows the expansion basis to be defined just once, that is only on the standard region. All other elements then can be considered as the image of the standard element under a parametric mapping. Consequently, the elemental operations of integration and differentiation can all be executed on the standard element, subject to a proper treatment of the transformation from local (world space) to standard (reference space) coordinates. For curved-sided elements, the mapping from standard element to local element is generally done using an *iso-parametric* representation. In this case, the local geometry is represented with an expansion of the same form and polynomial order as the unknown variables.

This structure, supplemented with building blocks such as block matrix linear algebra routines and automatic data coordinating objects, can be encapsulated in an efficient object-oriented C++ implementation.

This conceptual approach of the software leads to a high user-flexibility, including the selection of the preferred expansion basis, its polynomial order and the preferred numerical quadrature.

The website of the Nektar++ project can be found on: http://www.nektar.info

- Versioned with code
- User guide
  - Installation
  - Solver and Utility usage
  - Examples
  - Input file reference
  - FAQs
- Code documentation
  - Doxygen
  - Detailed implementation specifics

Support

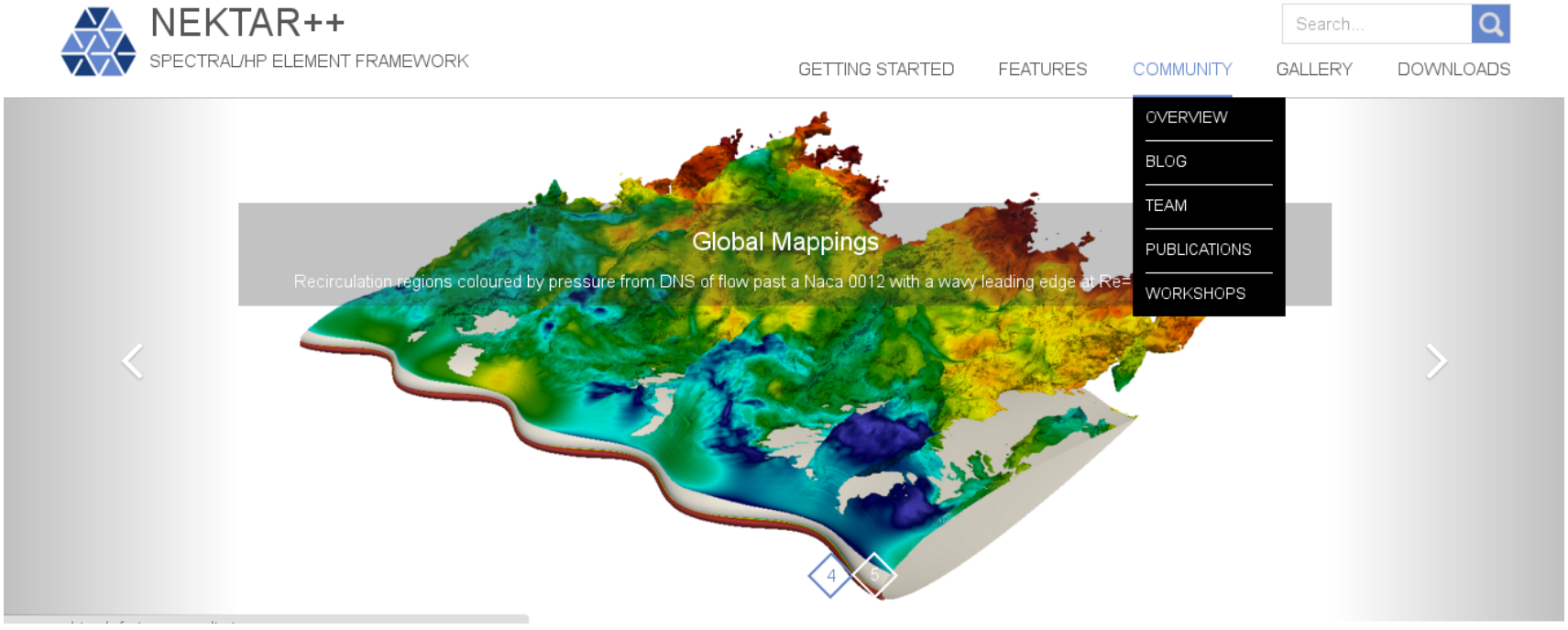
We are striving to build a community around Nektar++

- Support new users
- Make it easier for users to contribute back to the project.

Support

We are striving to build a community around Nektar++

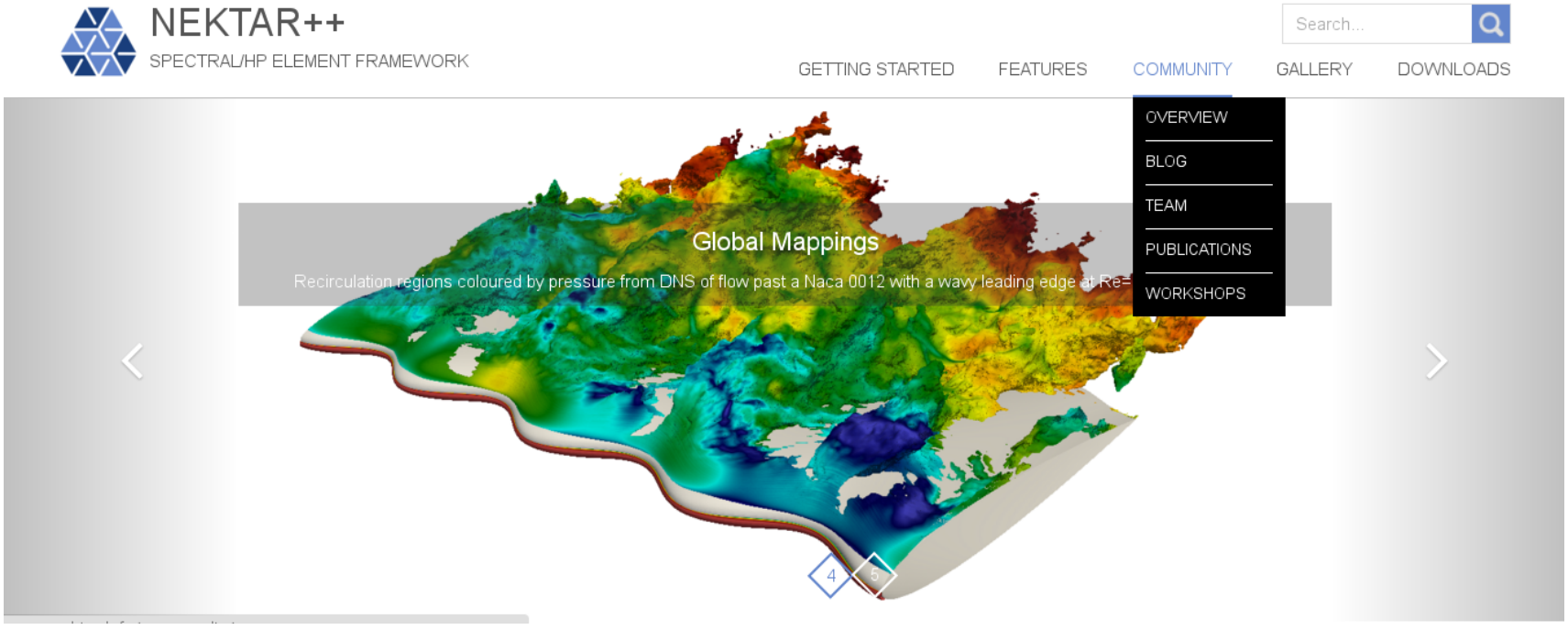
- Support new users
- Make it easier for users to contribute back to the project.
- New community-driven website



Support

We are striving to build a community around Nektar++

- Support new users
- Make it easier for users to contribute back to the project.
- New community-driven website



- Nektar++ Workshops (now an annual event!)
- Publication list
  - Help us demonstrate the impact of Nektar++
  - We can promote your research

Maintaining Code Stability

Nektar++ is a large project and therefore requires careful management of code changes to maintain stability.

We have implemented a range of tools and best practices to help facilitate this.

In the past year, there have been three key changes:

- Code review process
- Release model
- Change log and contribution guide

These build on a range of development practices we have already implemented.

Maintaining Code Stability

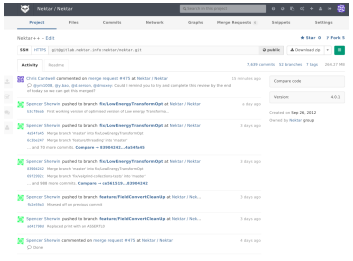
What development practices support a large multi-plaform collaborative software project such as Nektar++?

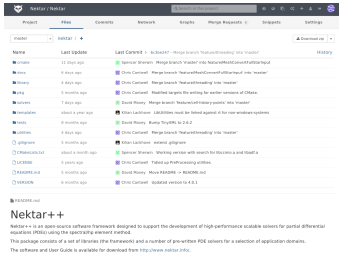
Maintaining Code Stability

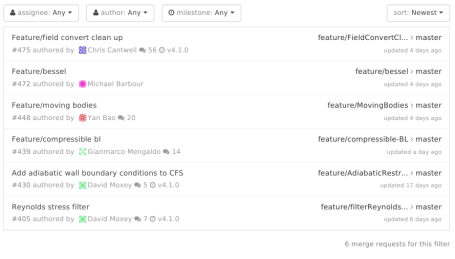
What development practices support a large multi-plaform collaborative software project such as Nektar++?

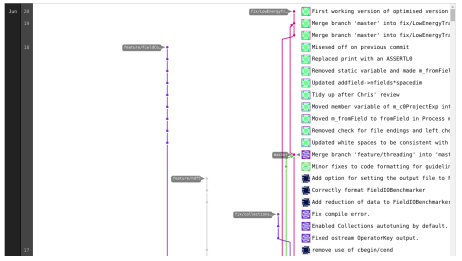
- Version-control (Git + GitLab)
- Issue tracking (GitLab)

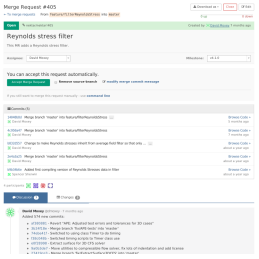
http://gitlab.nektar.info







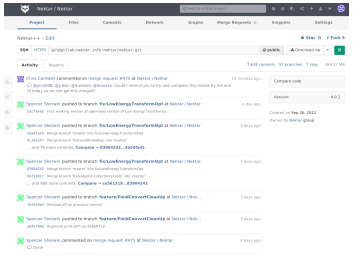


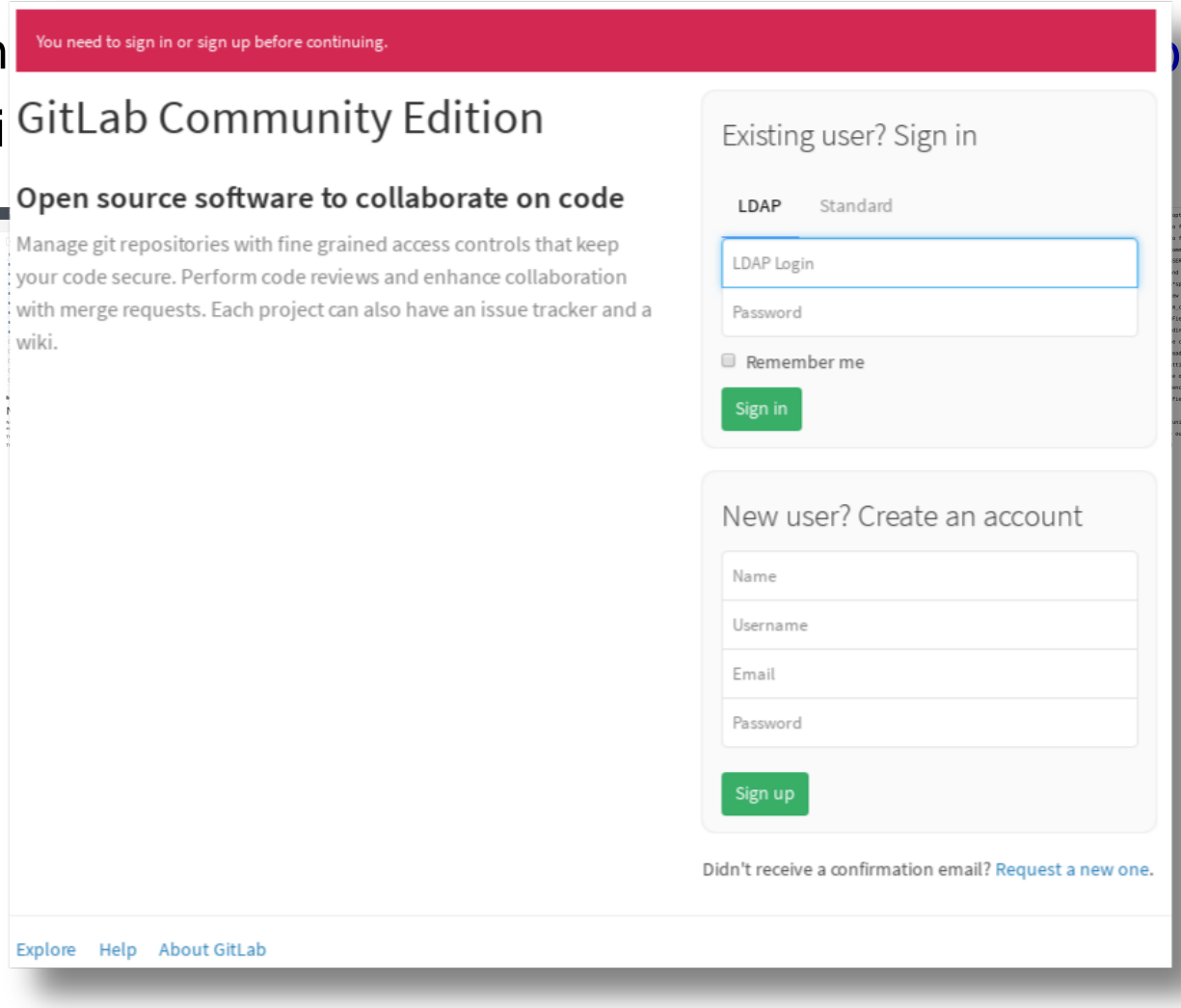


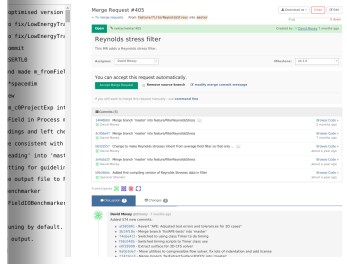
Maintaining Code Stability

What development practices support a large multi-plaform collaborative software project such as Nektar++?

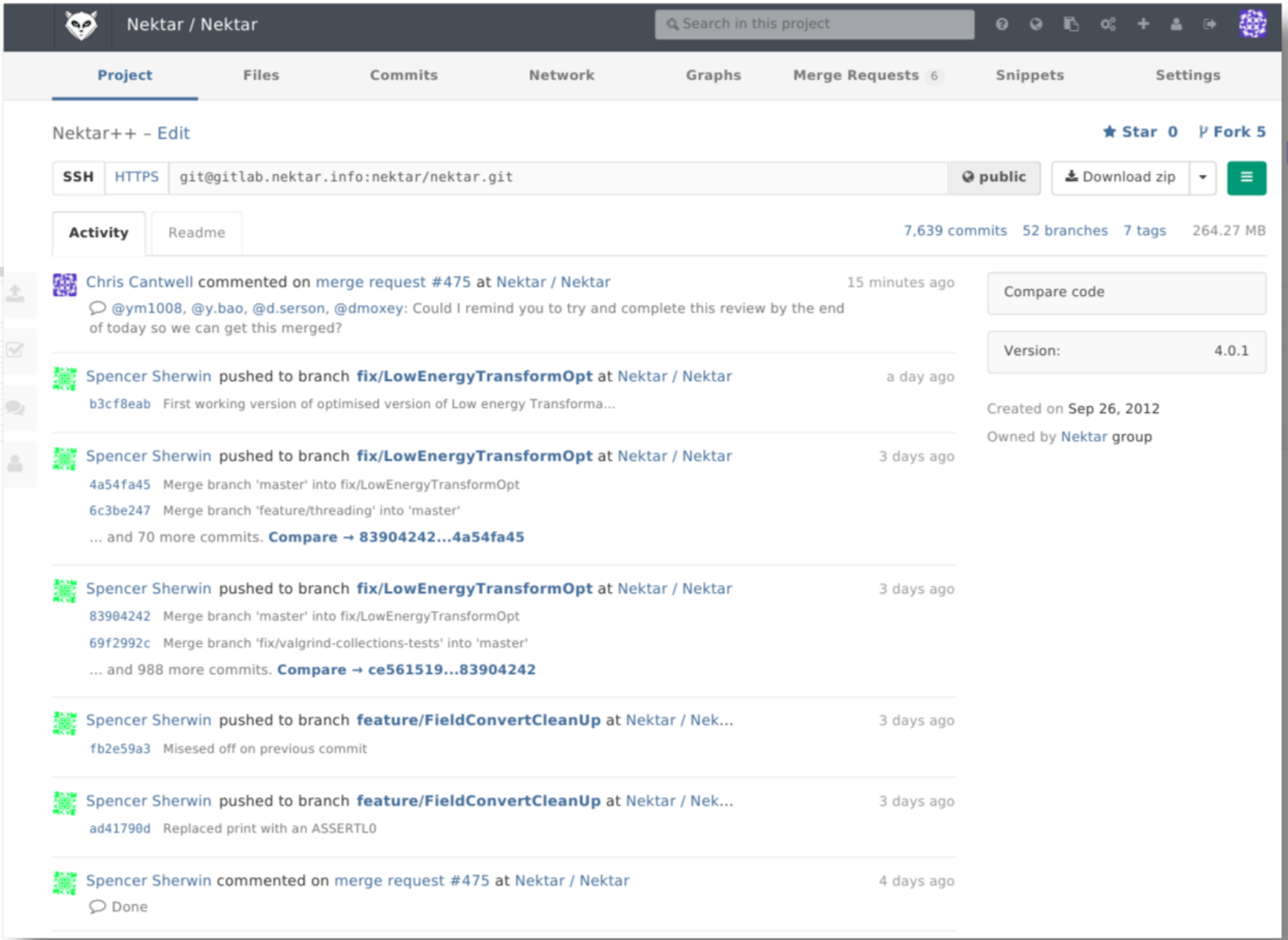
- Version-con .nektar.info
- Issue tracki



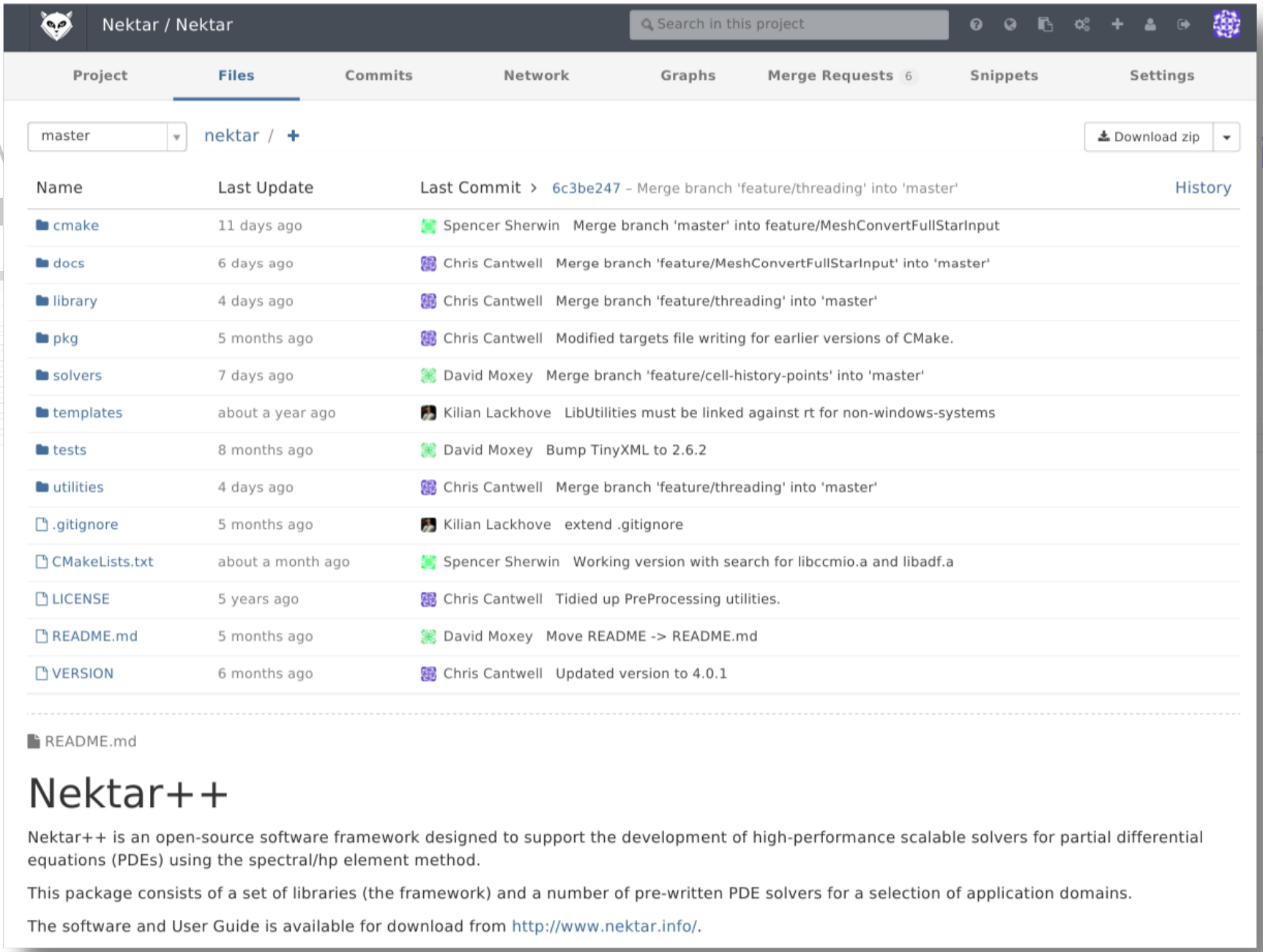




Maintaining Code Stability



Maintaining Code Stability



Nektar / Nektar

Search in this project

Project | Files | Commits | Network | Graphs | Merge Requests 6 | Snippets | Settings

master | nektar / + | Download zip

Name	Last Update	Last Commit › 6c3be247 – Merge branch 'feature/threading' into 'master'	History
cmake	11 days ago	Spencer Sherwin Merge branch 'master' into feature/MeshConvertFullStarInput	
docs	6 days ago	Chris Cantwell Merge branch 'feature/MeshConvertFullStarInput' into 'master'	
library	4 days ago	Chris Cantwell Merge branch 'feature/threading' into 'master'	
pkg	5 months ago	Chris Cantwell Modified targets file writing for earlier versions of CMake.	
solvers	7 days ago	David Moxey Merge branch 'feature/cell-history-points' into 'master'	
templates	about a year ago	Kilian Lackhove LibUtilities must be linked against rt for non-windows-systems	
tests	8 months ago	David Moxey Bump TinyXML to 2.6.2	
utilities	4 days ago	Chris Cantwell Merge branch 'feature/threading' into 'master'	
.gitignore	5 months ago	Kilian Lackhove extend .gitignore	
CMakeLists.txt	about a month ago	Spencer Sherwin Working version with search for libccmio.a and libadf.a	
LICENSE	5 years ago	Chris Cantwell Tidied up PreProcessing utilities.	
README.md	5 months ago	David Moxey Move README -> README.md	
VERSION	6 months ago	Chris Cantwell Updated version to 4.0.1	

README.md

Nektar++

Nektar++ is an open-source software framework designed to support the development of high-performance scalable solvers for partial differential equations (PDEs) using the spectral/hp element method.

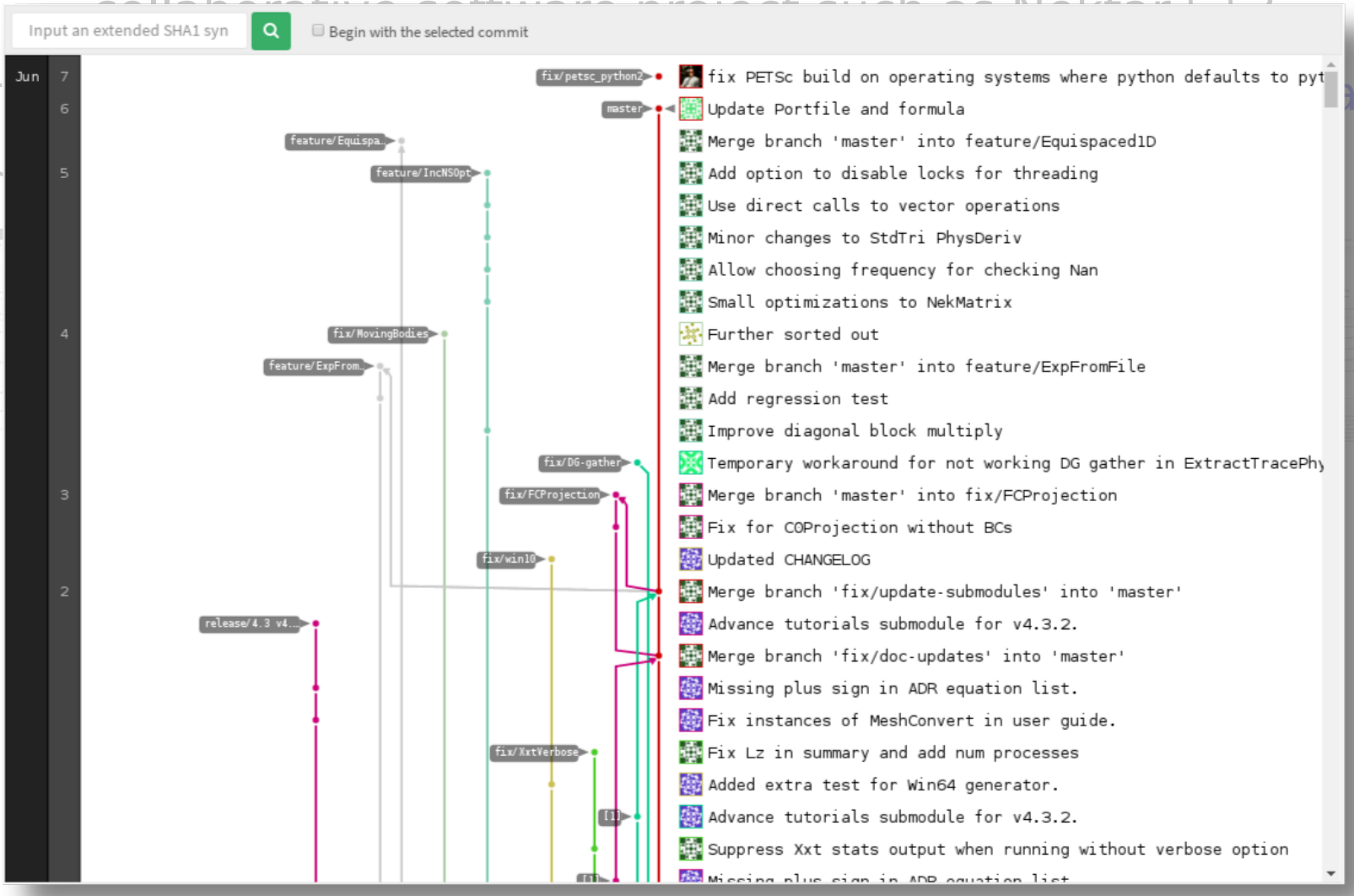
This package consists of a set of libraries (the framework) and a number of pre-written PDE solvers for a selection of application domains.

The software and User Guide is available for download from http://www.nektar.info/.

Maintaining Code Stability

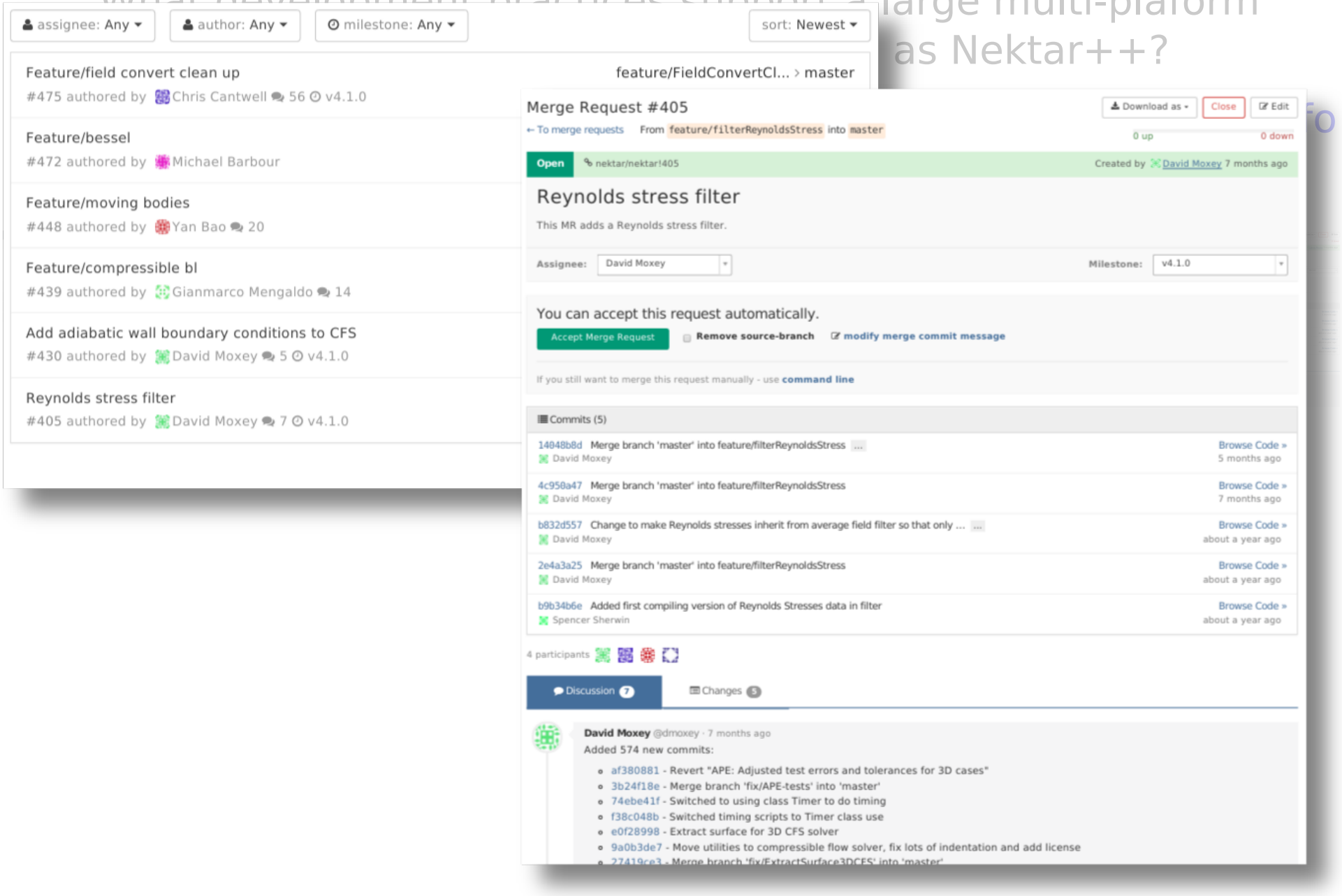
What development practices support a large multi-plaform collaborative software project such as Nektar++?

- Ver
- Issu



Maintaining Code Stability

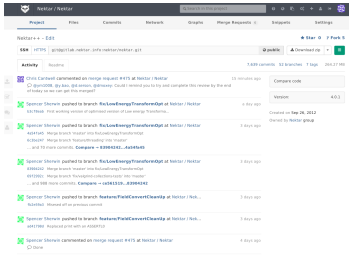
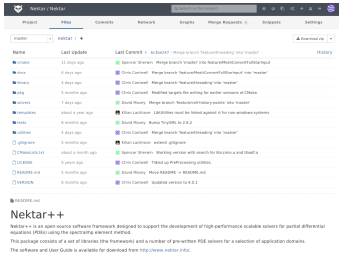
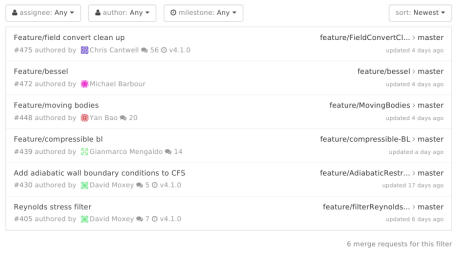
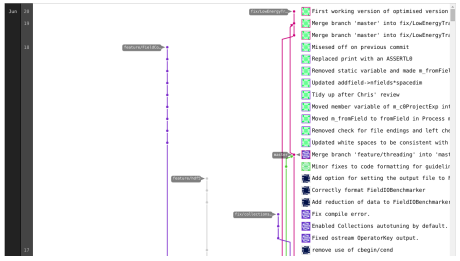
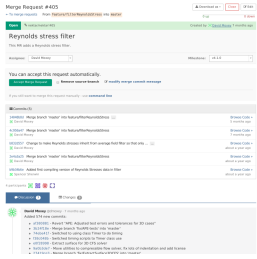
What development practices support a large multi-plaform as Nektar++?



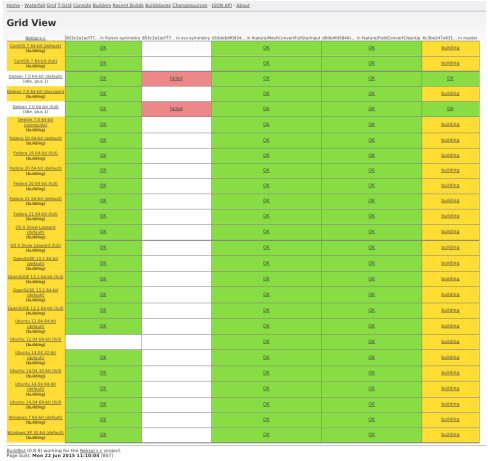
Maintaining Code Stability

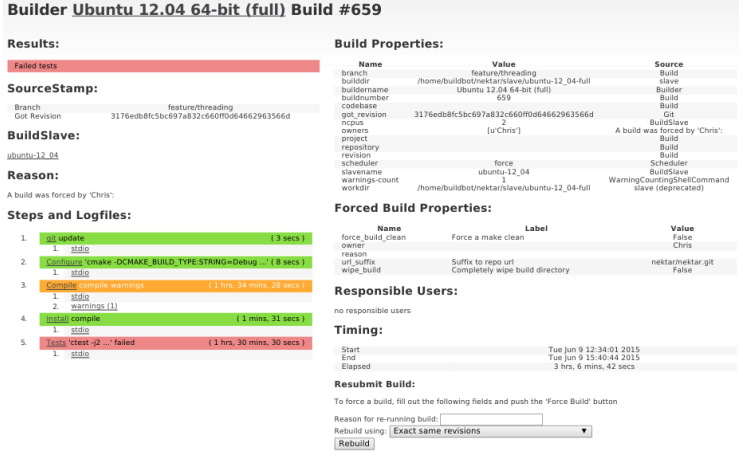
What development practices support a large multi-plaform collaborative software project such as Nektar++?

- Version-control (Git + GitLab) http://gitlab.nektar.info
- Issue tracking (GitLab)

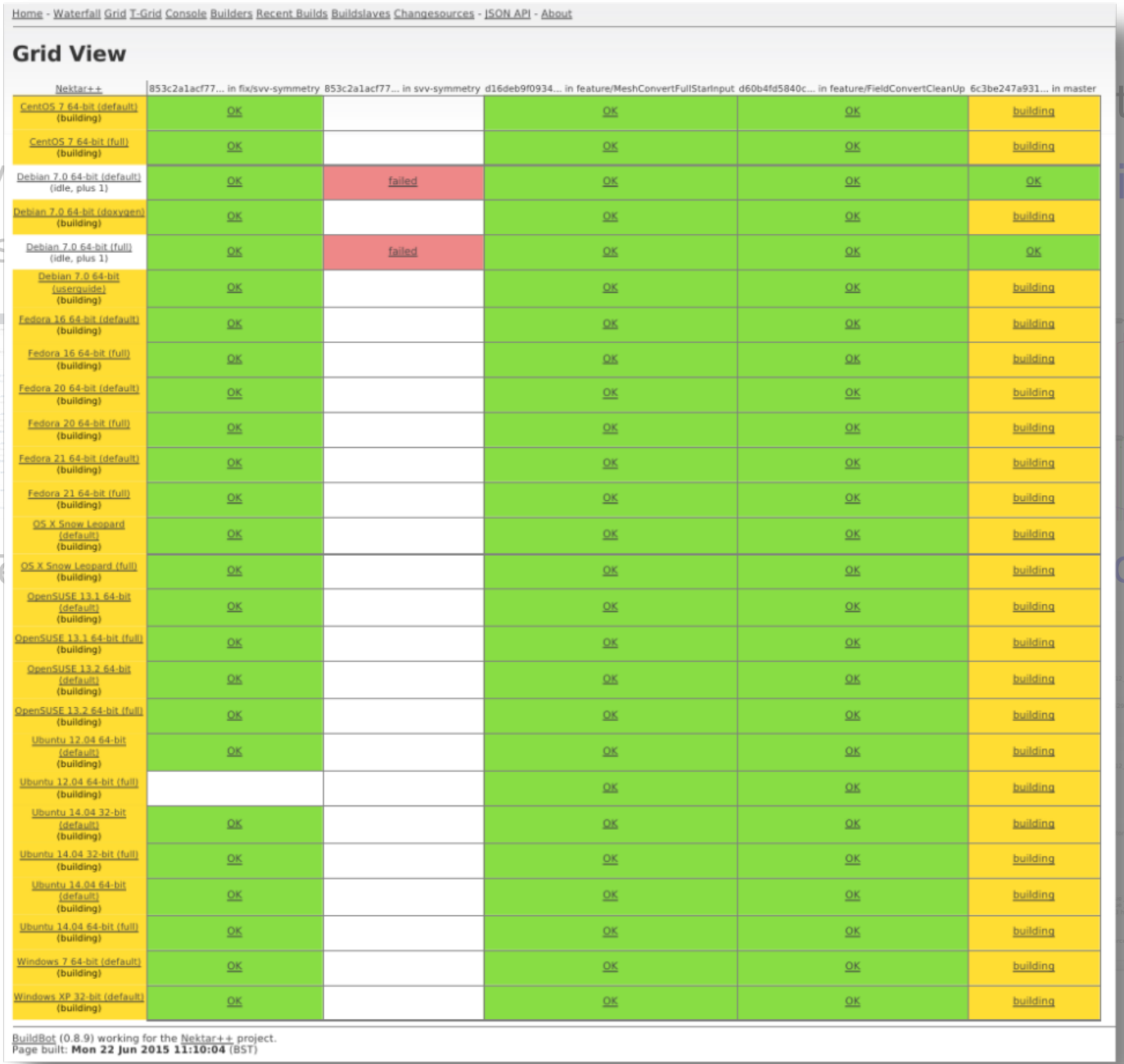






- Tests & Continuous Integration (Buildbot) http://buildbot.nektar.info





Maintaining Code Stability



Home - Waterfall Grid T-Grid Console Builders Recent Builds Buildslaves Changesources - JSON API - About

Grid View

Nektar++	853c2a1acf77... in fix/svv-symmetry	853c2a1acf77... in svv-symmetry	d16deb9f0934... in feature/MeshConvertFullStarInput	d60b4fd5840c... in feature/FieldConvertCleanUp	6c3be247a931... in master
CentOS 7 64-bit (default) (building)	OK		OK	OK	building
CentOS 7 64-bit (full) (building)	OK		OK	OK	building
Debian 7.0 64-bit (default) (idle, plus 1)	OK	failed	OK	OK	OK
Debian 7.0 64-bit (doxygen) (building)	OK		OK	OK	building
Debian 7.0 64-bit (full) (idle, plus 1)	OK	failed	OK	OK	OK
Debian 7.0 64-bit (userguide) (building)	OK		OK	OK	building
Fedora 16 64-bit (default) (building)	OK		OK	OK	building
Fedora 16 64-bit (full) (building)	OK		OK	OK	building
Fedora 20 64-bit (default) (building)	OK		OK	OK	building
Fedora 20 64-bit (full) (building)	OK		OK	OK	building
Fedora 21 64-bit (default) (building)	OK		OK	OK	building
Fedora 21 64-bit (full) (building)	OK		OK	OK	building
OS X Snow Leopard (default) (building)	OK		OK	OK	building
OS X Snow Leopard (full) (building)	OK		OK	OK	building
OpenSUSE 13.1 64-bit (default) (building)	OK		OK	OK	building
OpenSUSE 13.1 64-bit (full) (building)	OK		OK	OK	building
OpenSUSE 13.2 64-bit (default) (building)	OK		OK	OK	building
OpenSUSE 13.2 64-bit (full) (building)	OK		OK	OK	building
Ubuntu 12.04 64-bit (default) (building)	OK		OK	OK	building
Ubuntu 12.04 64-bit (full) (building)			OK	OK	building
Ubuntu 14.04 32-bit (default) (building)	OK		OK	OK	building
Ubuntu 14.04 32-bit (full) (building)	OK		OK	OK	building
Ubuntu 14.04 64-bit (default) (building)	OK		OK	OK	building
Ubuntu 14.04 64-bit (full) (building)	OK		OK	OK	building
Windows 7 64-bit (default) (building)	OK		OK	OK	building
Windows XP 32-bit (default) (building)	OK		OK	OK	building

BuildBot (0.8.9) working for the Nektar++ project.
Page built: **Mon 22 Jun 2015 11:10:04** (BST)

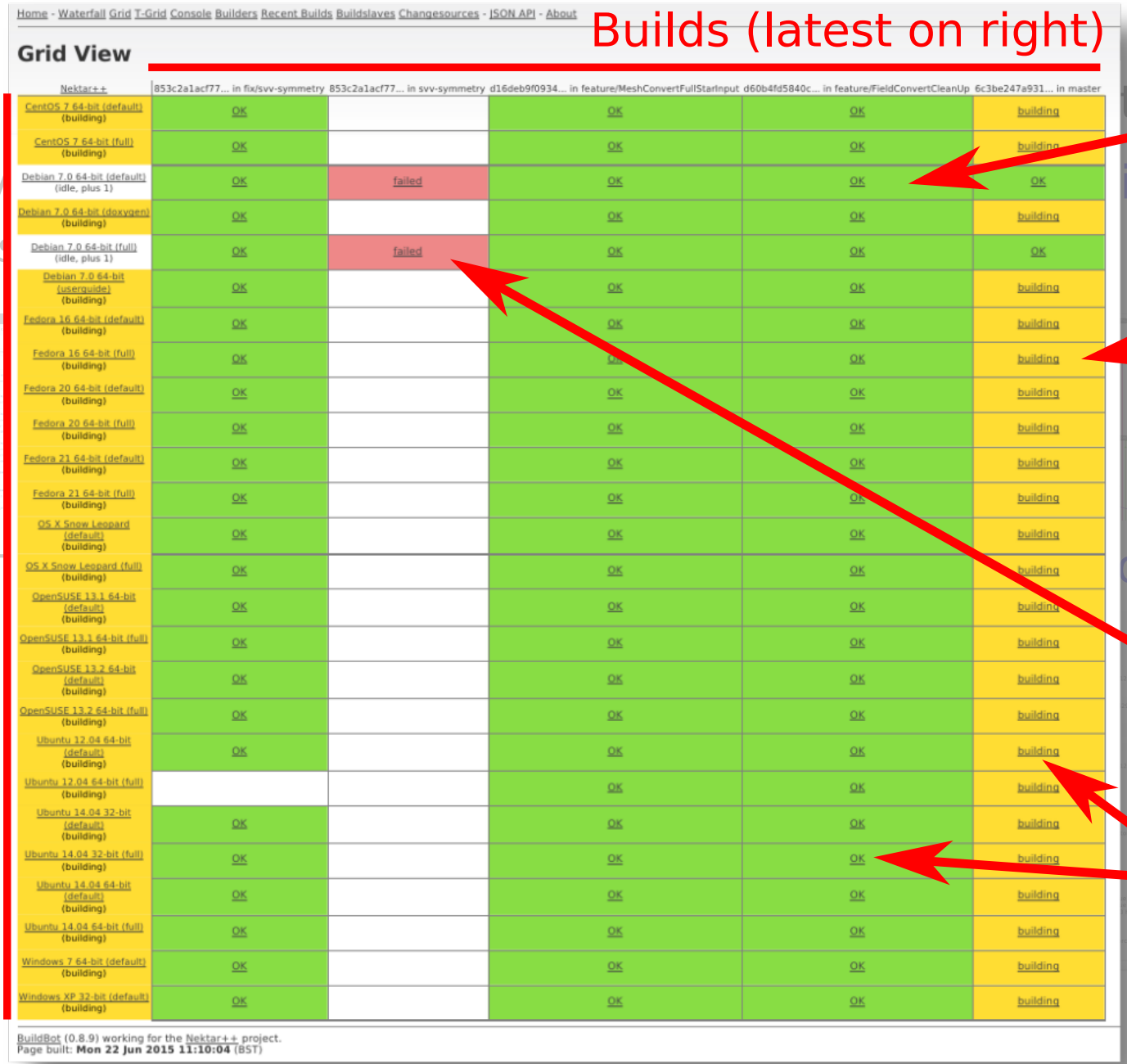
ulti-plaform

tar++?

itlab.nektar.info

dbot.nektar.info

Maintaining Code Stability



Maintaining Code Stability

Builder Ubuntu 12.04 64-bit (full) Build #659

Results:

Failed tests

SourceStamp:

Branch	feature/threading
Got Revision	3176edb8fc5bc697a832c660ff0d64662963566d

BuildSlave:

ubuntu-12_04

Reason:

A build was forced by 'Chris':

Steps and Logfiles:

1. git update (3 secs)
   1. stdio
2. Configure 'cmake -DCMAKE_BUILD_TYPE:STRING=Debug ...' (8 secs)
   1. stdio
3. Compile compile warnings (1 hrs, 34 mins, 28 secs)
   1. stdio
   2. warnings (1)
4. Install compile (1 mins, 31 secs)
   1. stdio
5. Tests 'ctest -j2 ...' failed (1 hrs, 30 mins, 30 secs)
   1. stdio

Build Properties:

Name	Value	Source
branch	feature/threading	Build
builddir	/home/buildbot/nektar/slave/ubuntu-12_04-full	slave
buildername	Ubuntu 12.04 64-bit (full)	Builder
buildnumber	659	Build
codebase		Build
got_revision	3176edb8fc5bc697a832c660ff0d64662963566d	Git
ncpus	2	BuildSlave
owners	[u'Chris']	A build was forced by 'Chris':
project		Build
repository		Build
revision		Build
scheduler	force	Scheduler
slavename	ubuntu-12_04	BuildSlave
warnings-count	1	WarningCountingShellCommand
workdir	/home/buildbot/nektar/slave/ubuntu-12_04-full	slave (deprecated)

Forced Build Properties:

Name	Label	Value
force_build_clean	Force a make clean	False
owner		Chris
reason		
url_suffix	Suffix to repo url	nektar/nektar.git
wipe_build	Completely wipe build directory	False

Responsible Users:

no responsible users

Timing:

Start	Tue Jun 9 12:34:01 2015
End	Tue Jun 9 15:40:44 2015
Elapsed	3 hrs, 6 mins, 42 secs

Resubmit Build:

To force a build, fill out the following fields and push the 'Force Build' button

Reason for re-running build:

Rebuild using: Exact same revisions

Rebuild

Maintaining Code Stability

Builder Ubuntu 12.04 64-bit (full) Build #659

Results:

Failed tests

SourceStamp:

Branch	feature/threading
Got Revision	3176edb8fc5bc697a832c660ff0d64662963566d

BuildSlave:

ubuntu-12_04

Reason:

A build was forced by 'Chris':

Steps and Logfiles:

1. git update
   1. stdio
2. Configure 'cmake -DCMAKE_BUILD_TYPE:STRING=Debug ...
   1. stdio
3. Compile compile warnings (1 hrs, 34 mins
   1. stdio
   2. warnings (1)
4. Install compile (1 mins
   1. stdio
5. Tests 'ctest -j2 ...' failed (1 hrs, 30 mins
   1. stdio

Build Properties:

Name	Value	Source
branch	feature/threading	Build
builddir	/home/buildbot/nektar/slave/ubuntu-12_04-full	slave
buildername	Ubuntu 12.04 64-bit (full)	Builder
buildnumber	659	Build
codebase		Build
got_revision	3176edb8fc5bc697a832c660ff0d64662963566d	Git
ncpus	2	BuildSlave
owners	[u'Chris']	A build was forced by 'Chris':
project		Build
repository		Build
revision		Build

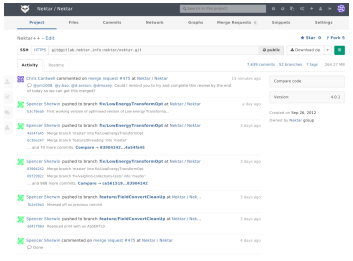
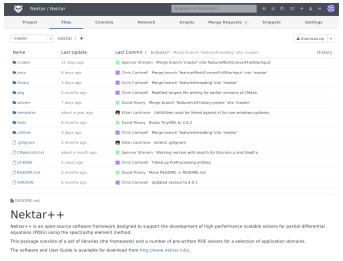
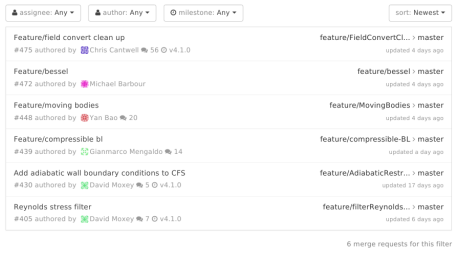
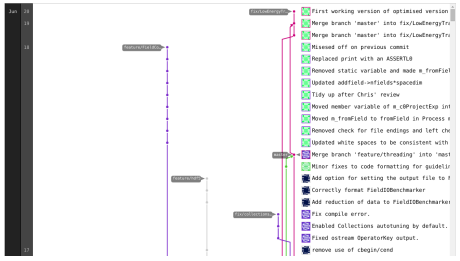
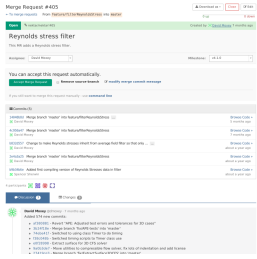
(view as text)

```
ctest -j2 '--timeout 600' --output-on-failure
 in dir /home/buildbot/nektar/slave/ubuntu-12_04-full/build/builds (timeout 1200 secs)
 watching logfiles {}
 argv: ['ctest', '-j2', '--timeout 600', '--output-on-failure']
 environment:
  HOME=/home/buildbot
  LANG=en_GB.UTF-8
  LANGUAGE=en_GB:en
  LOGNAME=buildbot
  MAIL=/var/mail/buildbot
  PATH=/opt/local/bin:/opt/local/lib/openmpi/bin:/usr/lib/openmpi/bin:/usr/lib64/openmpi/bin:/usr/lib64/mpi/gcc/openmpi/bin
  PWD=/home/buildbot/nektar/slave/ubuntu-12_04-full/build/builds
  SHELL=/bin/sh
  TERM=linux
  USER=buildbot
  XDG_SESSION_COOKIE=204222634a5a369277e2004500000003-1423328811.240864-1156323198
 using PTY: False
Test project /home/buildbot/nektar/slave/ubuntu-12_04-full/build/builds
        Start 437: APESolver_APE_3DPulse_WeakDG_MODIFIED
        Start 438: APESolver_APE_3DPulseWall_WeakDG_MODIFIED
  1/463 Test #438: APESolver_APE_3DPulseWall_WeakDG_MODIFIED .....   Passed  201.29 sec
        Start 163: IncNavierStokesSolver_bfs_tg
  2/463 Test #437: APESolver_APE_3DPulse_WeakDG_MODIFIED .........   Passed  349.90 sec
        Start 188: IncNavierStokesSolver_Cyl_AdaptiveSFD
  3/463 Test #163: IncNavierStokesSolver_bfs_tg ..................   Passed  152.24 sec
        Start 459: MeshConvert_StarTec/StraightRW
  4/463 Test #459: MeshConvert_StarTec/StraightRW ................   Passed  123.47 sec
        Start 202: IncNavierStokesSolver_bfs_tg_par
  5/463 Test #188: IncNavierStokesSolver_Cyl_AdaptiveSFD .........   Passed  131.09 sec
        Start 252: ADRSolver_Advection3D_m12_DG_hex_VarP
  6/463 Test #252: ADRSolver_Advection3D_m12_DG_hex_VarP .........   Passed   89.24 sec
        Start 357: ADRSolver_Advection3D_m12_DG_hex_VarP_par
  7/463 Test #202: IncNavierStokesSolver_bfs_tg_par ..............   Passed  113.37 sec
        Start 190: IncNavierStokesSolver_ChanFlow_3DH2D_FFT
  8/463 Test #357: ADRSolver_Advection3D_m12_DG_hex_VarP_par .....   Passed   75.62 sec
        Start 148: IncNavierStokesSolver_ChanFlow_3DH2D_MVM
  9/463 Test #190: IncNavierStokesSolver_ChanFlow_3DH2D_FFT ......   Passed   68.67 sec
        Start 144: IncNavierStokesSolver_ChanStability
 10/463 Test #144: IncNavierStokesSolver_ChanStability ...........   Passed   39.79 sec
        Start 435: APESolver_APE_2DPulseWall_WeakDG_MODIFIED
 11/463 Test #148: IncNavierStokesSolver_ChanFlow_3DH2D_MVM ......   Passed   66.52 sec
```

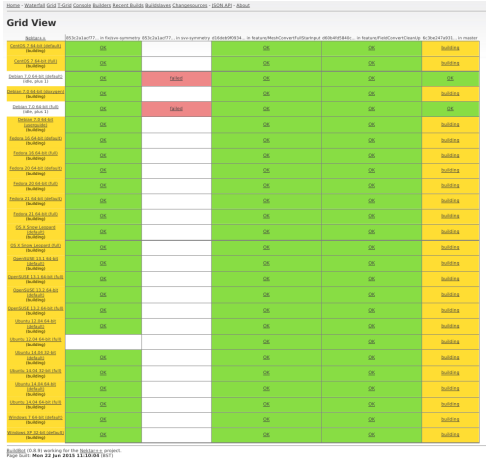
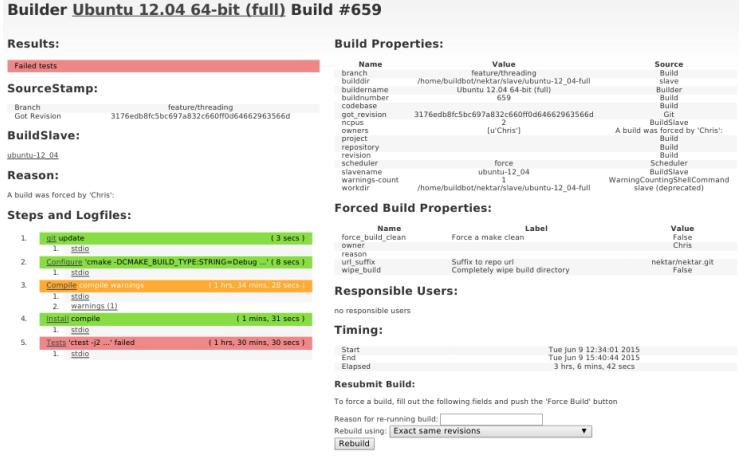
Maintaining Code Stability

What development practices support a large multi-plaform collaborative software project such as Nektar++?

- Version-control (Git + GitLab) http://gitlab.nektar.info
- Issue tracking (GitLab)

- Tests & Continuous Integration (Buildbot) http://buildbot.nektar.info

- Documentation (PDF + doxygen) http://doc.nektar.info

Release model

Previous model and problems:

- New features and fixes continuously merged into master
- Code releases every 6 months - 1 year
- New features constantly introducing instability

New approach:

- New features are continuously merged into master
- Bug fixes are additionally merged into a separate release branch
- More frequent *patch* releases
- Should remain stable and bug fixes are distributed quicker

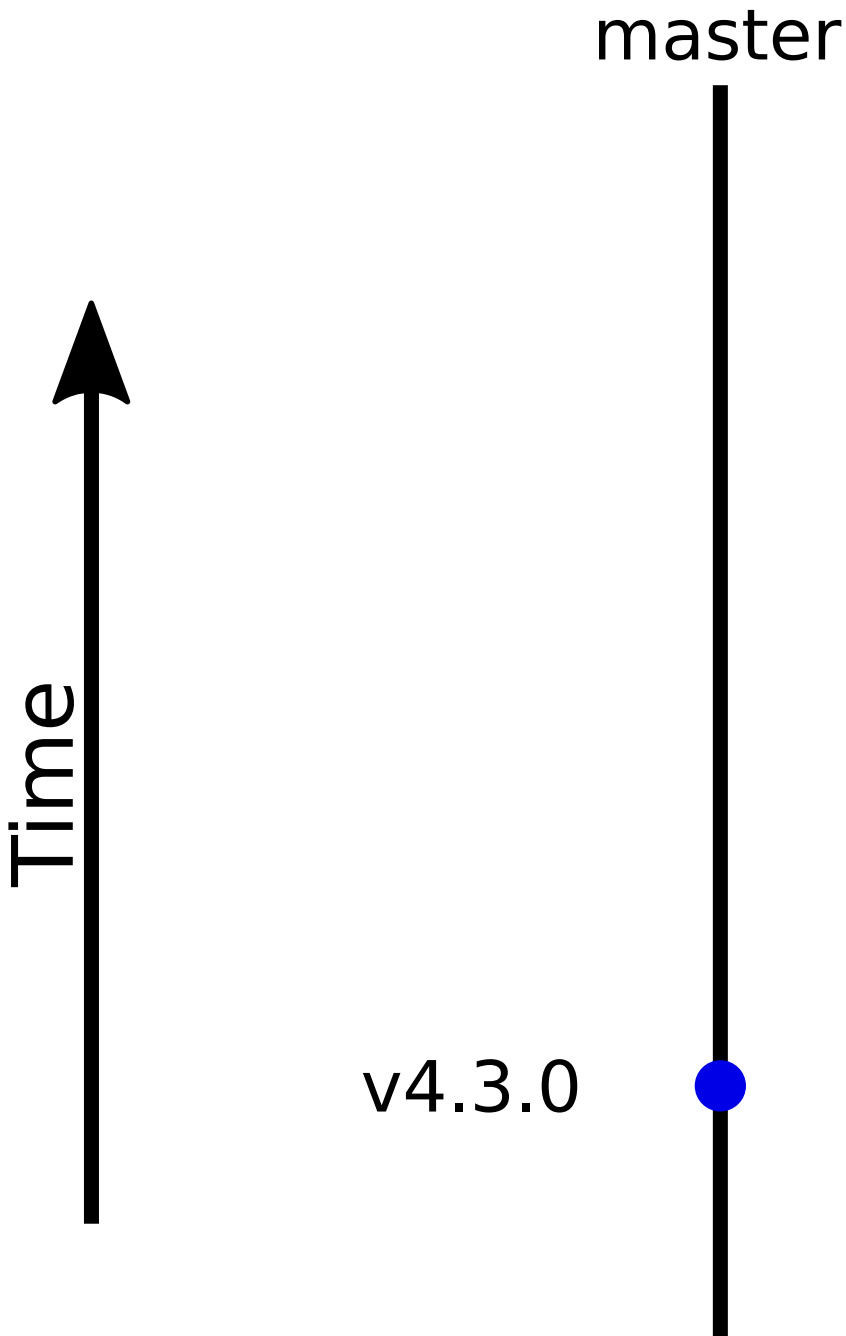
Release model

- Release version number

v4.3.1

Major . Minor . Patch

Release model



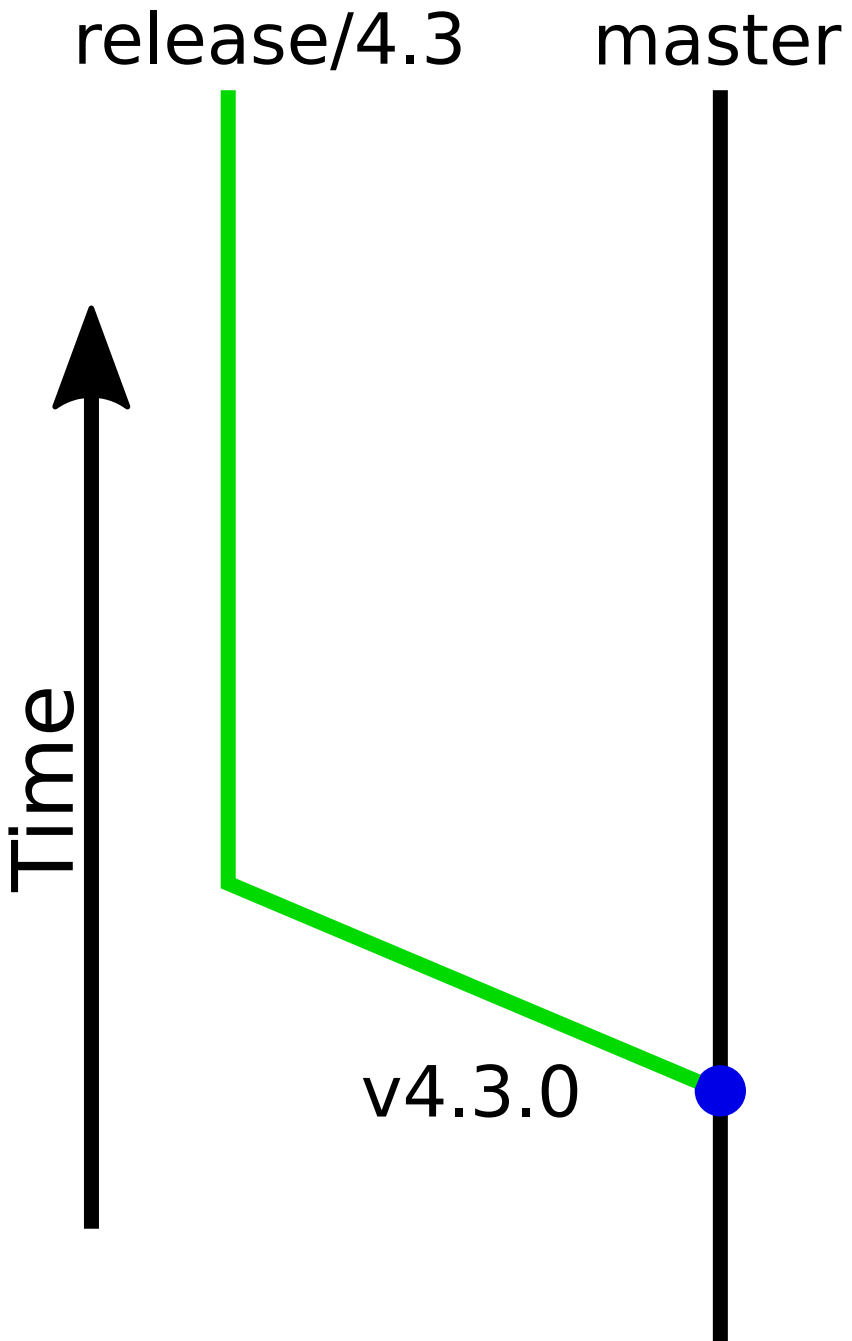
- Release version number

v4.3.1

Major . Minor . Patch

- Primary development in 'master'

Release model



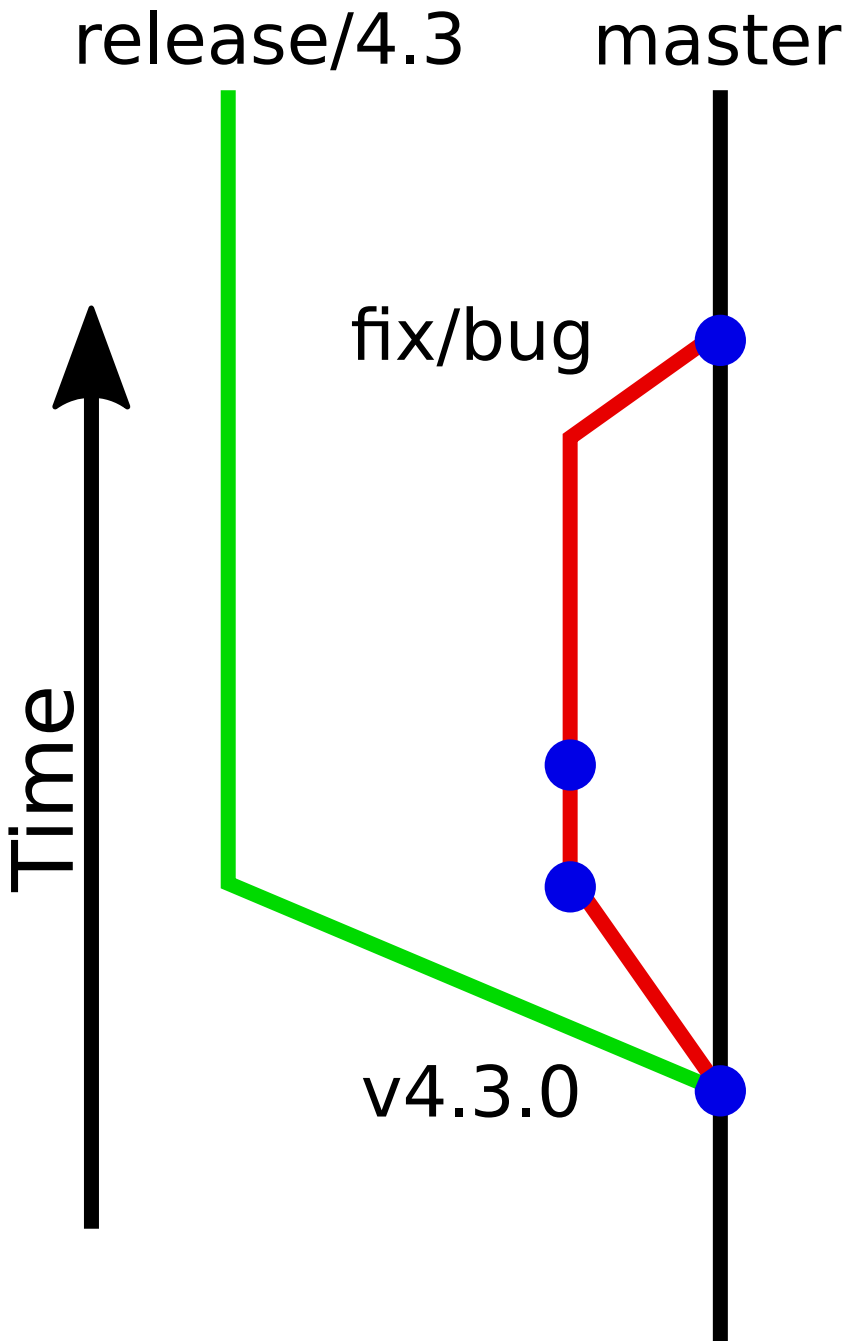
- Release version number

v4.3.1

Major . Minor . Patch

- Primary development in 'master'
- Major/Minor releases tagged in 'master'
- Create 'release' branch

Release model



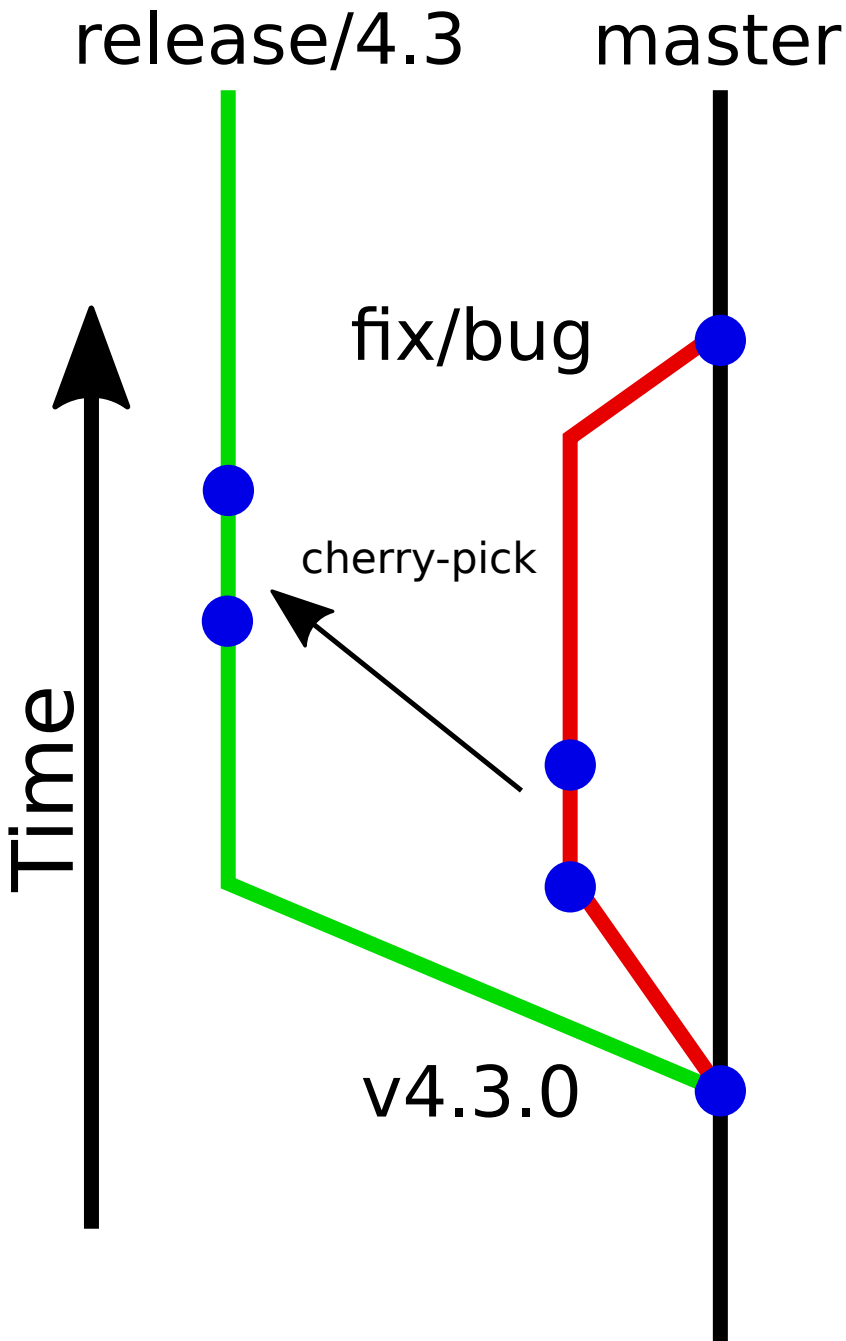
- Release version number

v4.3.1

Major . Minor . Patch

- Primary development in 'master'
- Major/Minor releases tagged in 'master'
- Create 'release' branch
- Bug-fixes applied in branch and merged into 'master' as normal

Release model



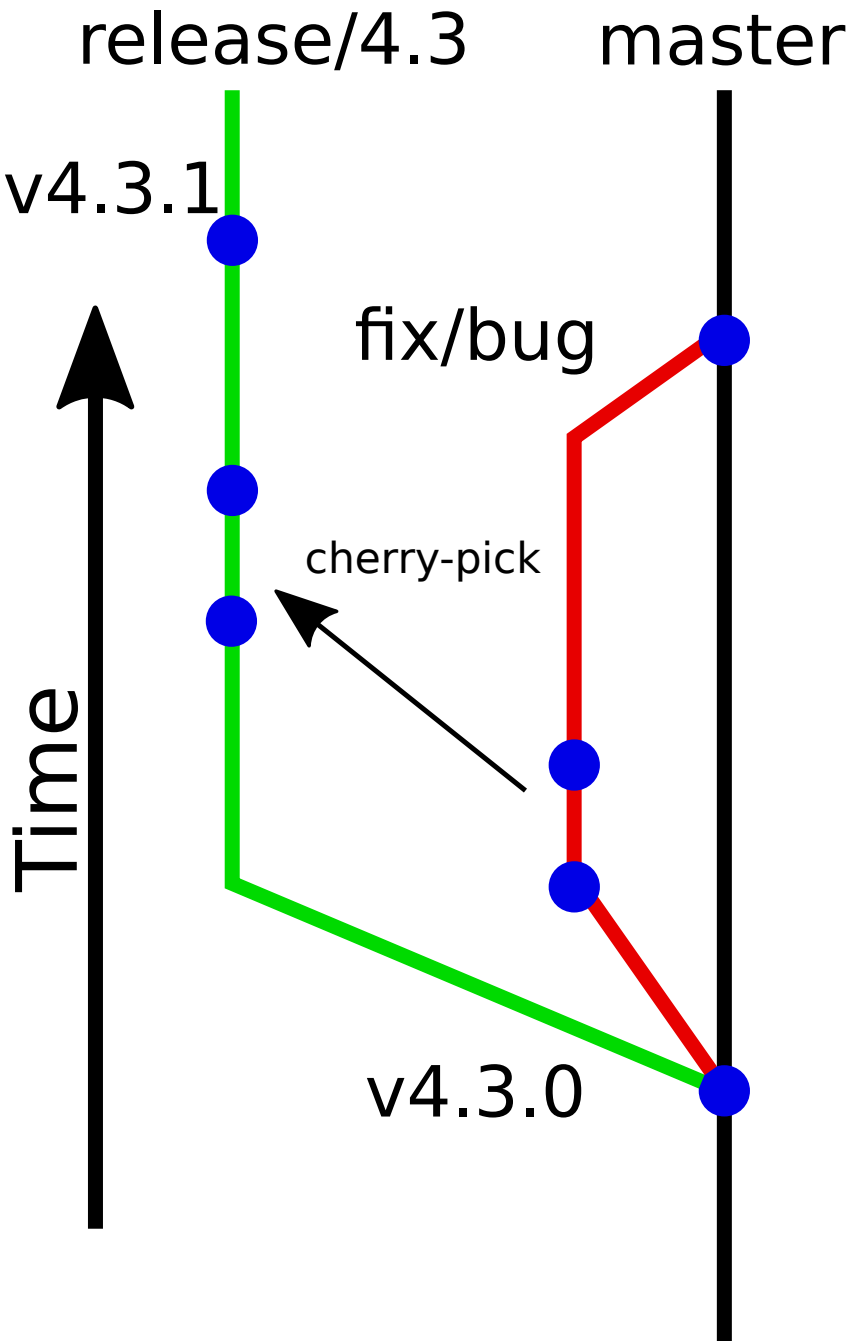
- Release version number

v4.3.1

Major . Minor . Patch

- Primary development in 'master'
- Major/Minor releases tagged in 'master'
- Create 'release' branch
- Bug-fixes applied in branch and merged into 'master' as normal
- Cherry-picked into 'release' branch

Release model



- Release version number

v4.3.1

Major . Minor . Patch

- Primary development in 'master'
- Major/Minor releases tagged in 'master'
- Create 'release' branch
- Bug-fixes applied in branch and merged into 'master' as normal
- Cherry-picked into 'release' branch
- Monthly patch releases

Review Process

Peer-reviewed (non-blinded) process is similar to that used in journals.

1. New code is committed to a branch in git
   - `feature/[name]` for new functionality
   - `fix/[name]` for bug fixes
   - `tidy/[name]` code formatting updates (no functional change)
   - `ticket/[number]-[description]` for resolving specific issues

Review Process

Peer-reviewed (non-blinded) process is similar to that used in journals.

1. New code is committed to a branch in git
   - `feature/[name]` for new functionality
   - `fix/[name]` for bug fixes
   - `tidy/[name]` code formatting updates (no functional change)
   - `ticket/[number]-[description]` for resolving specific issues
2. Author ensures code meets the *contribution criteria* and *code guidelines* for the project
   - Details about contribution criteria are in `CONTRIBUTING.md`

Review Process

Peer-reviewed (non-blinded) process is similar to that used in journals.

1. New code is committed to a branch in git
   - `feature/[name]` for new functionality
   - `fix/[name]` for bug fixes
   - `tidy/[name]` code formatting updates (no functional change)
   - `ticket/[number]-[description]` for resolving specific issues
2. Author ensures code meets the *contribution criteria* and *code guidelines* for the project
   - Details about contribution criteria are in `CONTRIBUTING.md`
3. Author submits a merge request on GitLab (http://gitlab.nektar.info)
4. A senior developer will oversee the review process and assign reviewers
   - Typically two reviewers for new features
   - Typically one reviewer for smaller fixes

Review Process

Peer-reviewed (non-blinded) process is similar to that used in journals.

1. New code is committed to a branch in git
   - `feature/[name]` for new functionality
   - `fix/[name]` for bug fixes
   - `tidy/[name]` code formatting updates (no functional change)
   - `ticket/[number]-[description]` for resolving specific issues
2. Author ensures code meets the *contribution criteria* and *code guidelines* for the project
   - Details about contribution criteria are in `CONTRIBUTING.md`
3. Author submits a merge request on GitLab (http://gitlab.nektar.info)
4. A senior developer will oversee the review process and assign reviewers
   - Typically two reviewers for new features
   - Typically one reviewer for smaller fixes
5. Reviewers suggest any changes or improvements, verify code meets *contribution criteria* and *code guidelines*, etc.
6. Iterate between authors and reviewers until everyone happy.
7. Senior developer performs the merge.

Review Process: Contribution criteria

Some of the key aspects required include:

- Documentation
  - Has new user-exposed features been documented in the user guide?
  - Have any relevant tutorials been updated?
  - Do new functions/classes include doxygen documentation?
  - Is the code sensibly commented?

- Testing
  - Have regression tests been added for new features?
  - Do existing regression tests all pass on buildbot?

- Formatting according to *coding guidelines*
  - 80 char width, indentation of 4 spaces, no tabs, no namespace indentation
  - All code blocks should use braces, braces on new lines
  - Avoid preprocessor directives where possible
  - Spaces around operators, after `if`, `while`, etc
- Appropriately updated the CHANGELOG

Further information, and What's Next?

Further information

- `http://www.nektar.info`
- *Nektar++: An open-source spectral/hp element framework*, C. D. Cantwell, D. Moxey, A. Comerford, *et al.*, Computer Physics Communications, vol.192, p205–219, 2015

What's Next?

- Expand tutorials
- Developer's Guide
- Packaged binaries for Windows