Imperial College London

# Nektar++: Library design

**Chris Cantwell**, David Moxey, Mike Kirby, Spencer Sherwin

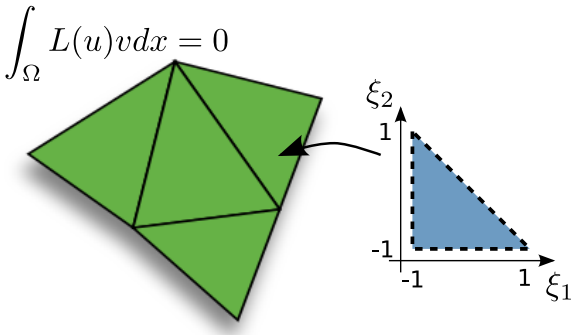
Nektar++ Workshop
Imperial College London
7th July 2015

NEKTAR++
SPECTRAL/HP ELEMENT FRAMEWORK

# Overview

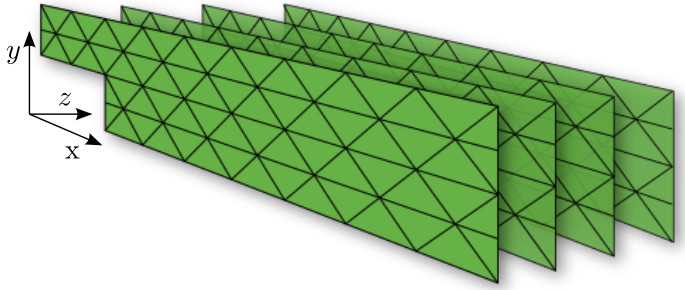
- Library design overview
- Collections
- Developer Practice
- User Guide and Developer Guide

# Nektar++: Objectives



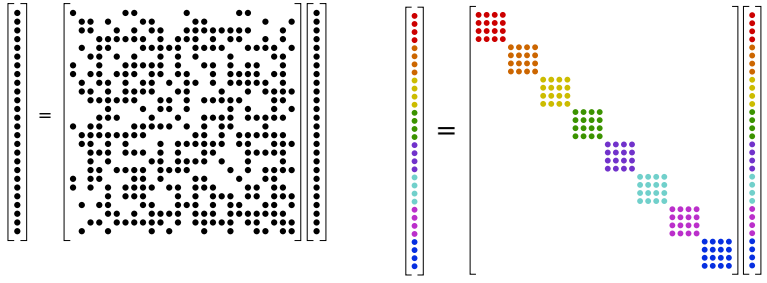

## Mathematical Construction

Expose different discretisations (CG, DG) by combining and reusing low-level elemental mathematical constructs.



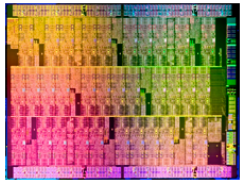

Cross-dimensional support: 1D, 2D and 3D

Retain and exploit domain symmetries and embeddings (homogeneous, cylindrical, manifold)



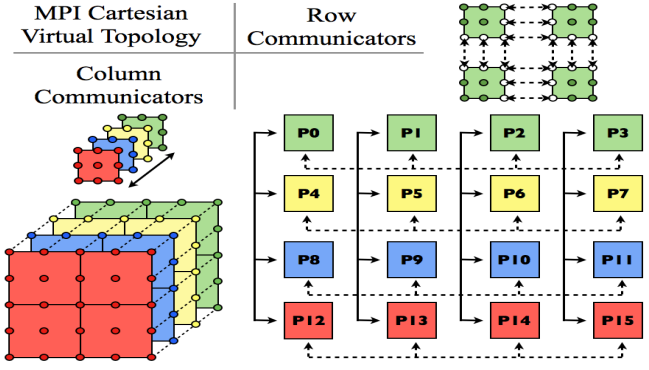
## Computational Implementation

Challenge high-/low-order boundaries while maintaining efficiency.

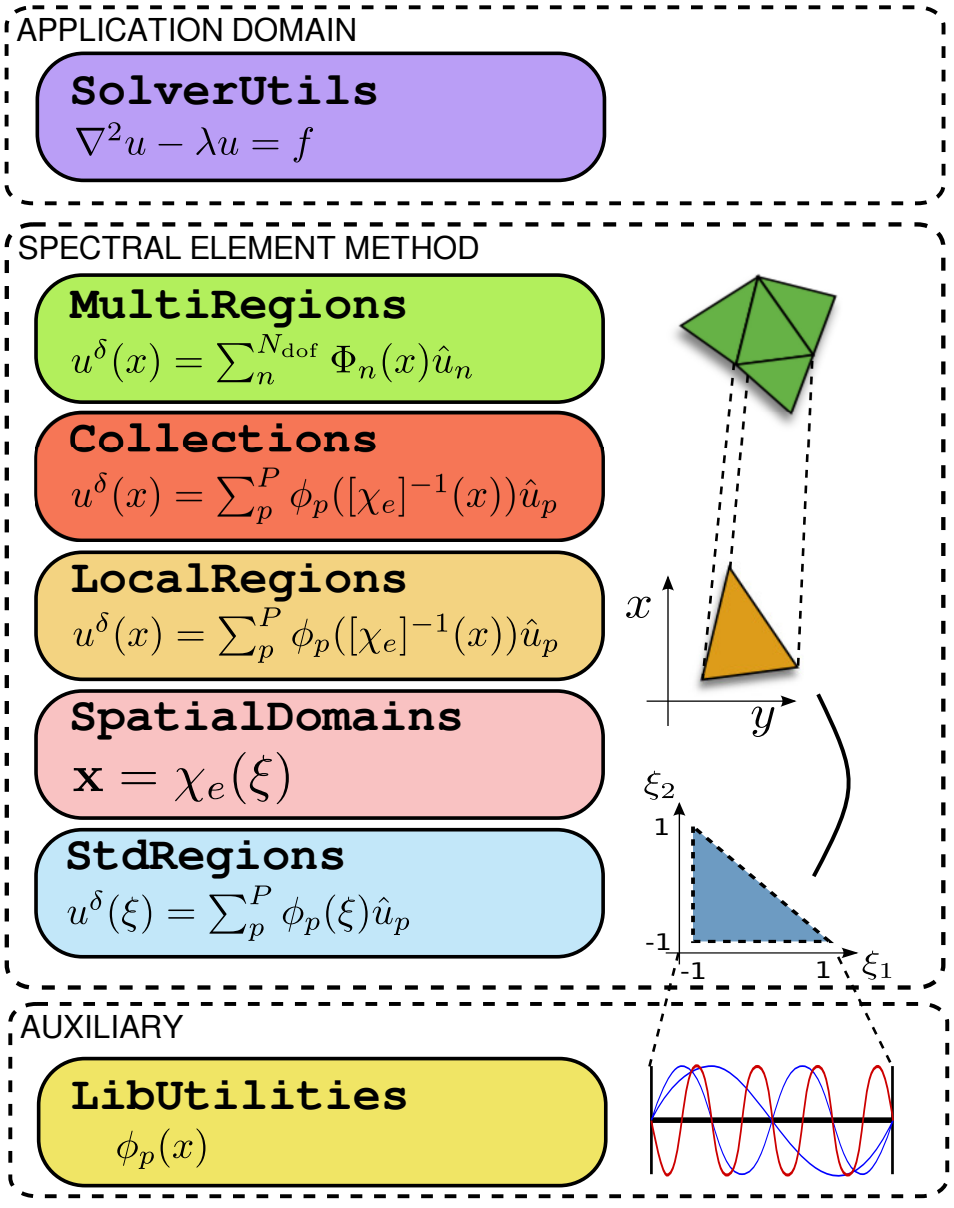



Bridge current and future hardware diversity.

Achieve flexible HPC scalability and performance through hybrid parallelism.

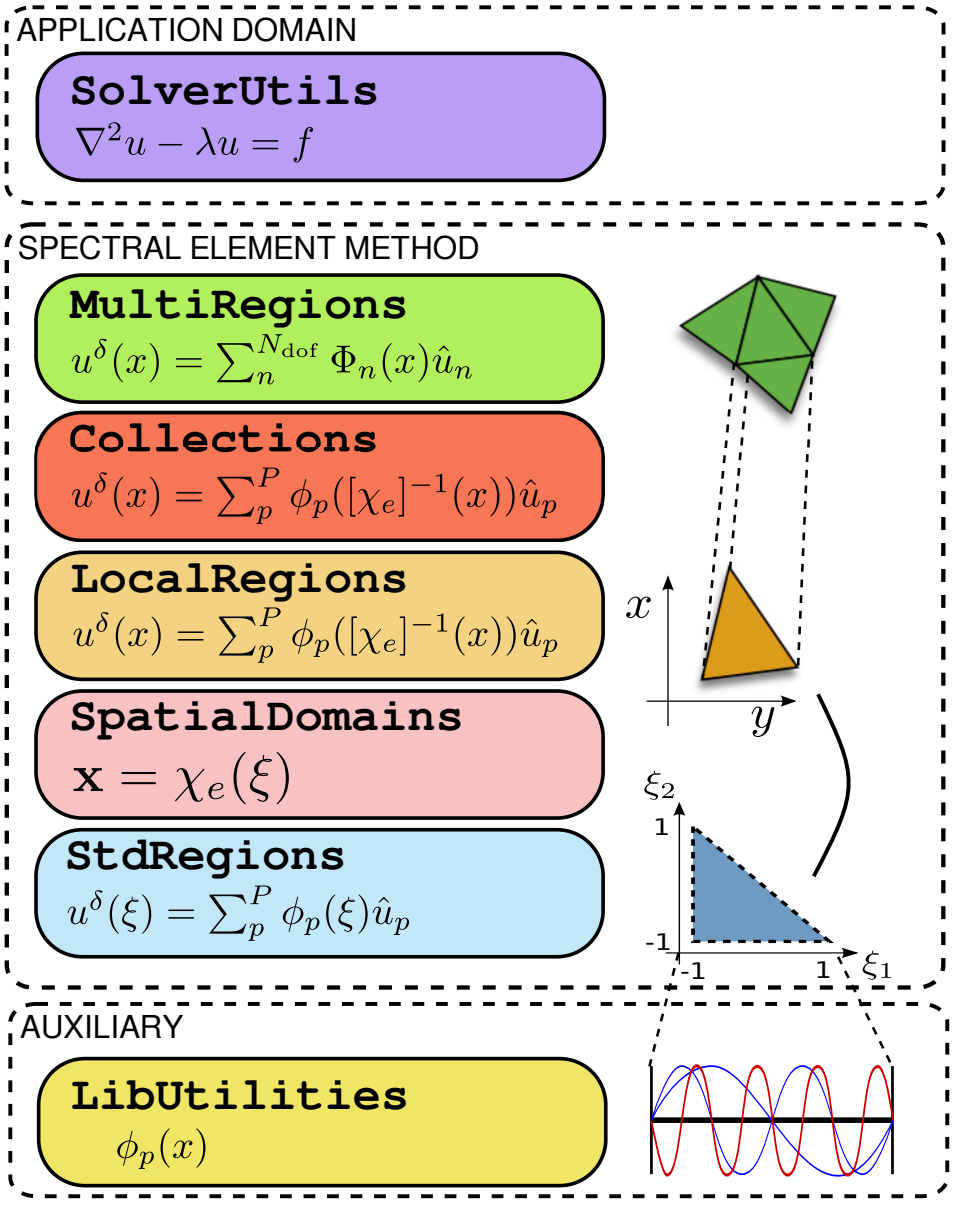


# Nektar++: Design



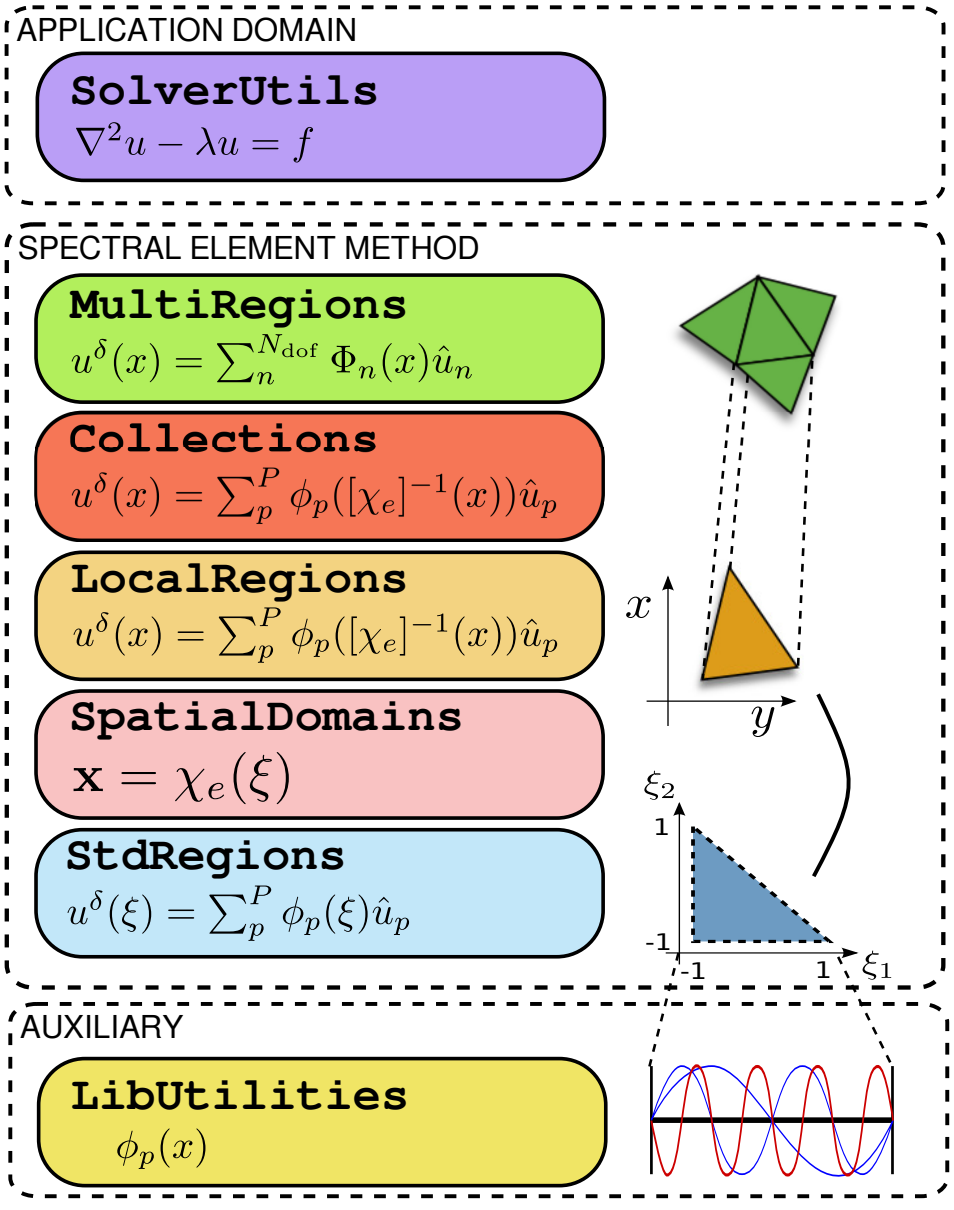

- Stack of libraries
- Build up the spectral/$hp$ element method systematically

# Nektar++: Design




- Stack of libraries
- Build up the spectral/$hp$ element method systematically
- Core mathematical abstractions at bottom
- Complete physical domain description and operators at top

# Nektar++: Design




- Stack of libraries
- Build up the spectral/$hp$ element method systematically
- Core mathematical abstractions at bottom
- Complete physical domain description and operators at top
- Encapsulate complexities of the spectral/$hp$ element formulation at the top levels to improve accessibility of methods
- Allow users to still engage at lower level depending on their requirements (e.g. for methods research)

# Nektar++: Operator construction

Nektar++ is a spectral/hp element toolkit written in C++.

Helmholtz problem: $\nabla^2 u + \lambda u = f$

# Nektar++: Operator construction

Nektar++ is a spectral/hp element toolkit written in C++.

Helmholtz problem: $$\nabla^2 u + \lambda u = f$$

$$\downarrow$$

Weak form + IBP: $$-(\nabla u, \nabla v) + \lambda(u, v) + (\nabla u, v)|_{d\Omega} = (f, v)$$

# Nektar++: Operator construction

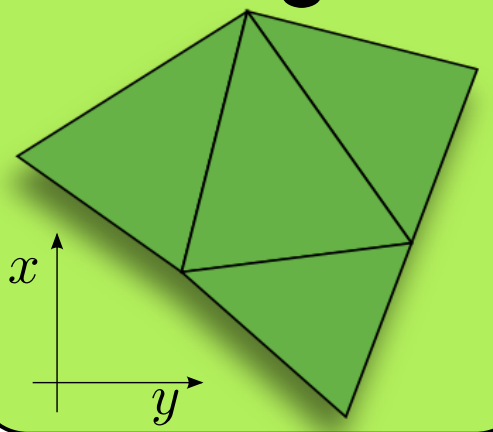
$$u^\delta = \sum_i \hat{u}_i \Phi_i(x)$$

Nektar++ is a spectral/hp element toolkit written in C++.

Helmholtz problem: $\nabla^2 u + \lambda u = f$

Weak form + IBP: $-(\nabla u, \nabla v) + \lambda(u, v) + (\nabla u, v)|_{d\Omega} = (f, v)$

**MultiRegions**




$$\mathbf{f}[i] = \int_\Omega \Phi_i(x) f(x) dx$$

# Nektar++: Operator construction

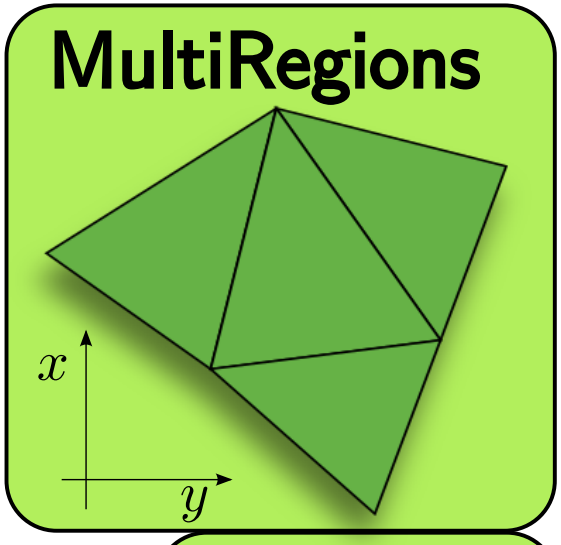
$$u^\delta = \sum_i \hat{u}_i \Phi_i(x)$$

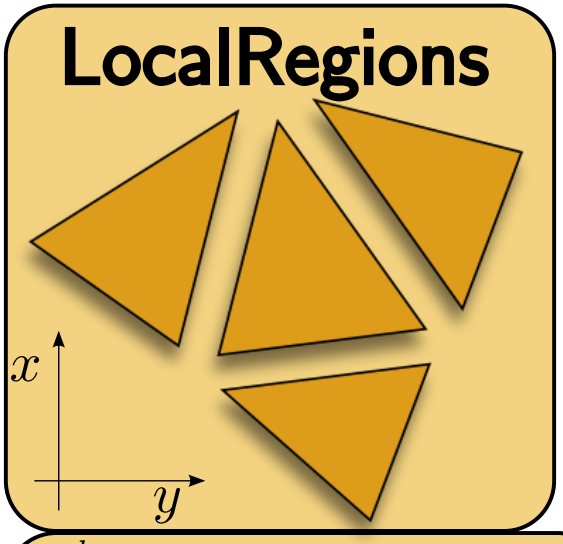
Nektar++ is a spectral/hp element toolkit written in C++.

$$u_e^\delta = \sum_p \hat{u}_p \phi_p(x)$$

Helmholtz problem: $\nabla^2 u + \lambda u = f$

Weak form + IBP: $-(\nabla u, \nabla v) + \lambda(u, v) + (\nabla u, v)|_{d\Omega} = (f, v)$




$$\mathbf{f}[i] = \int_\Omega \Phi_i(x) f(x) dx = \sum_e^{N^{el}} \sum_p \int_{\Omega^e} \phi_p(x) f(x) dx$$

# Nektar++: Operator construction

$$u^\delta = \sum_i \hat{u}_i \Phi_i(x)$$

$$u_e^\delta = \sum_p \hat{u}_p \phi_p(x)$$

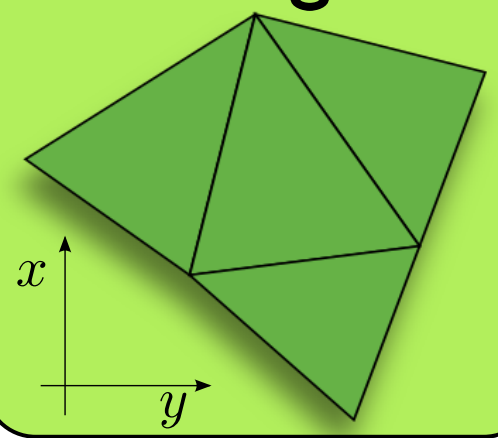
Nektar++ is a spectral/hp element toolkit written in C++.

Helmholtz problem: $\nabla^2 u + \lambda u = f$

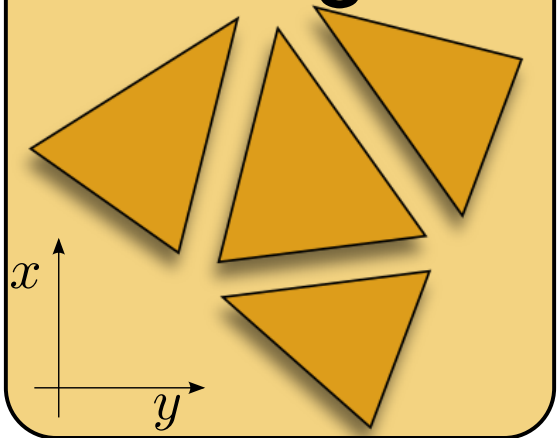
↓

Weak form + IBP: $-(\nabla u, \nabla v) + \lambda(u, v) + (\nabla u, v)|_{d\Omega} = (f, v)$

## MultiRegions



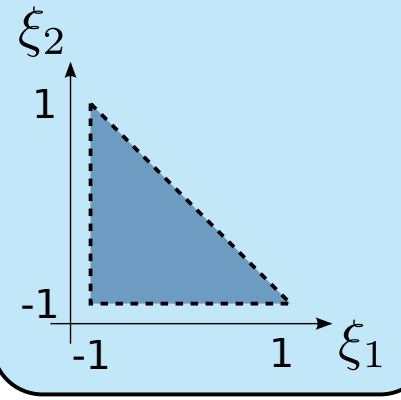

## LocalRegions




$$\mathbf{f}[i] = \int_\Omega \Phi_i(x) f(x) dx = \sum_e^{N^{el}} \sum_p \int_{\Omega^e} \phi_p(x) f(x) dx$$

$$= \sum_e \sum_p \int_{\Omega^e} \phi_p(\chi^e(\xi)) f(\chi^e(\xi)) J^e d\xi$$

## StdRegions



# Nektar++: Operator construction

$$u^\delta = \sum_i \hat{u}_i \Phi_i(x)$$

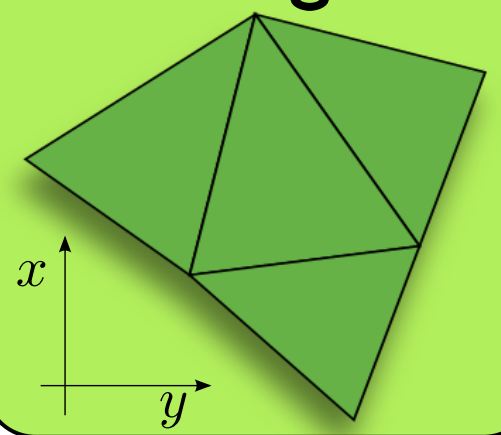
$$u_e^\delta = \sum_p \hat{u}_p \phi_p(x)$$

Nektar++ is a spectral/hp element toolkit written in C++.

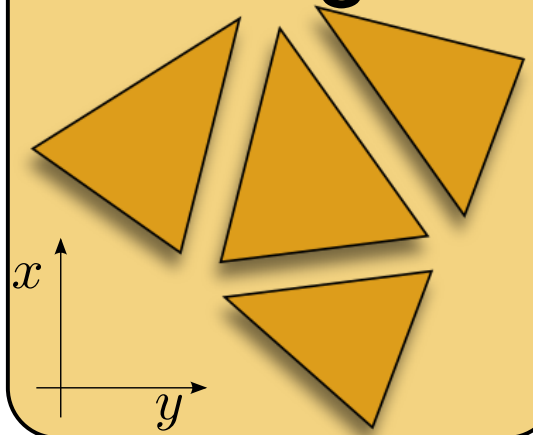
Helmholtz problem: $\nabla^2 u + \lambda u = f$

Weak form + IBP: $-(\nabla u, \nabla v) + \lambda(u, v) + (\nabla u, v)|_{d\Omega} = (f, v)$

## MultiRegions



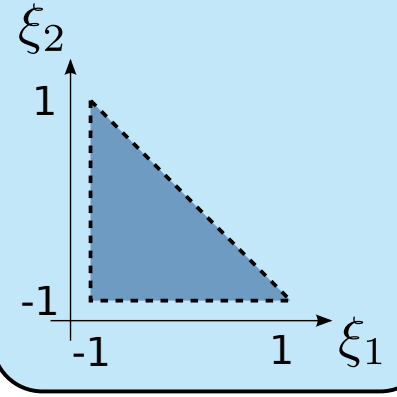

## LocalRegions




## SpatialDomains

$\chi^e(\xi)$

## StdRegions




$$\mathbf{f}[i] = \int_\Omega \Phi_i(x) f(x) dx = \sum_e^{N^{el}} \sum_p \int_{\Omega^e} \phi_p(x) f(x) dx$$

$$= \sum_e \sum_p \int_{\Omega^e} \phi_p(\chi^e(\xi)) f(\chi^e(\xi)) J^e d\xi$$

# Nektar++: Operator construction

$$u^\delta = \sum_i \hat{u}_i \Phi_i(x)$$

$$u_e^\delta = \sum_p \hat{u}_p \phi_p(x)$$

Nektar++ is a spectral/hp element toolkit written in C++.

Helmholtz problem: $\nabla^2 u + \lambda u = f$

Weak form + IBP: $-(\nabla u, \nabla v) + \lambda(u, v) + (\nabla u, v)|_{d\Omega} = (f, v)$

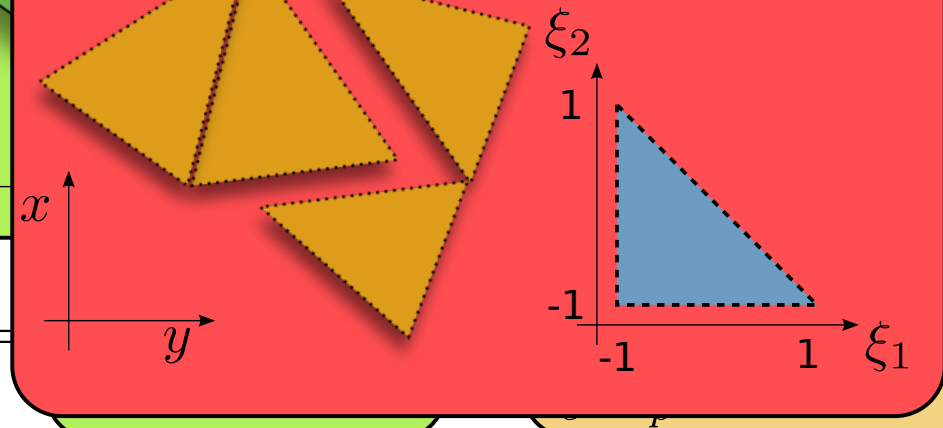
**MultiRegions**

**LocalRegions**

**SpatialDomains**

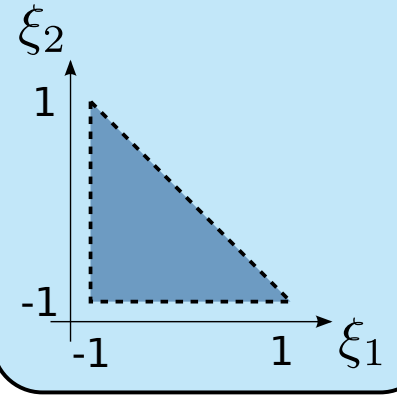
$\chi^e(\xi)$

**Collections**



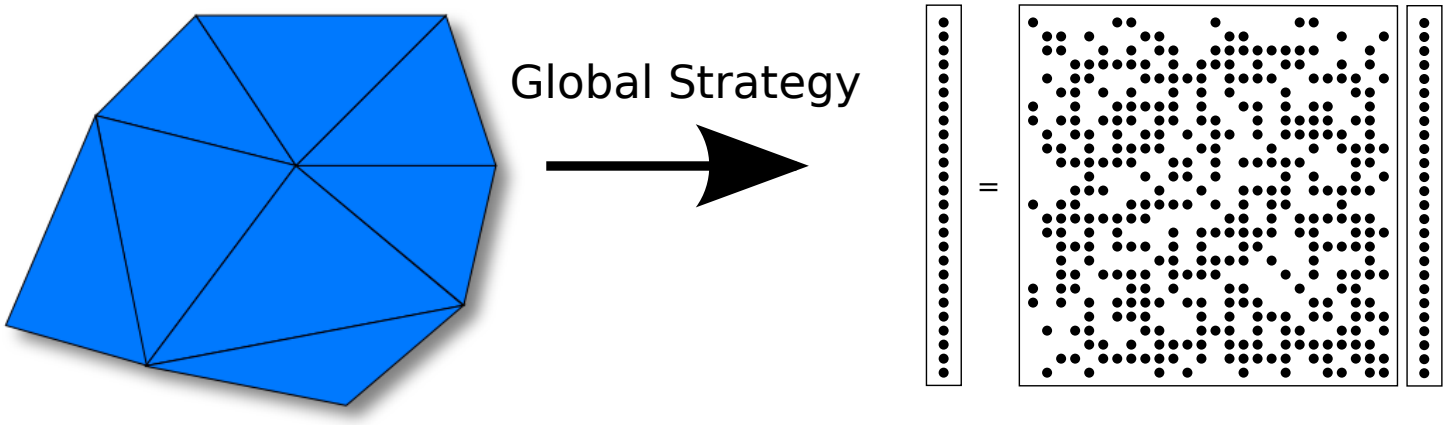

$\mathbf{f}[i] =$ $f(x)dx$

**StdRegions**

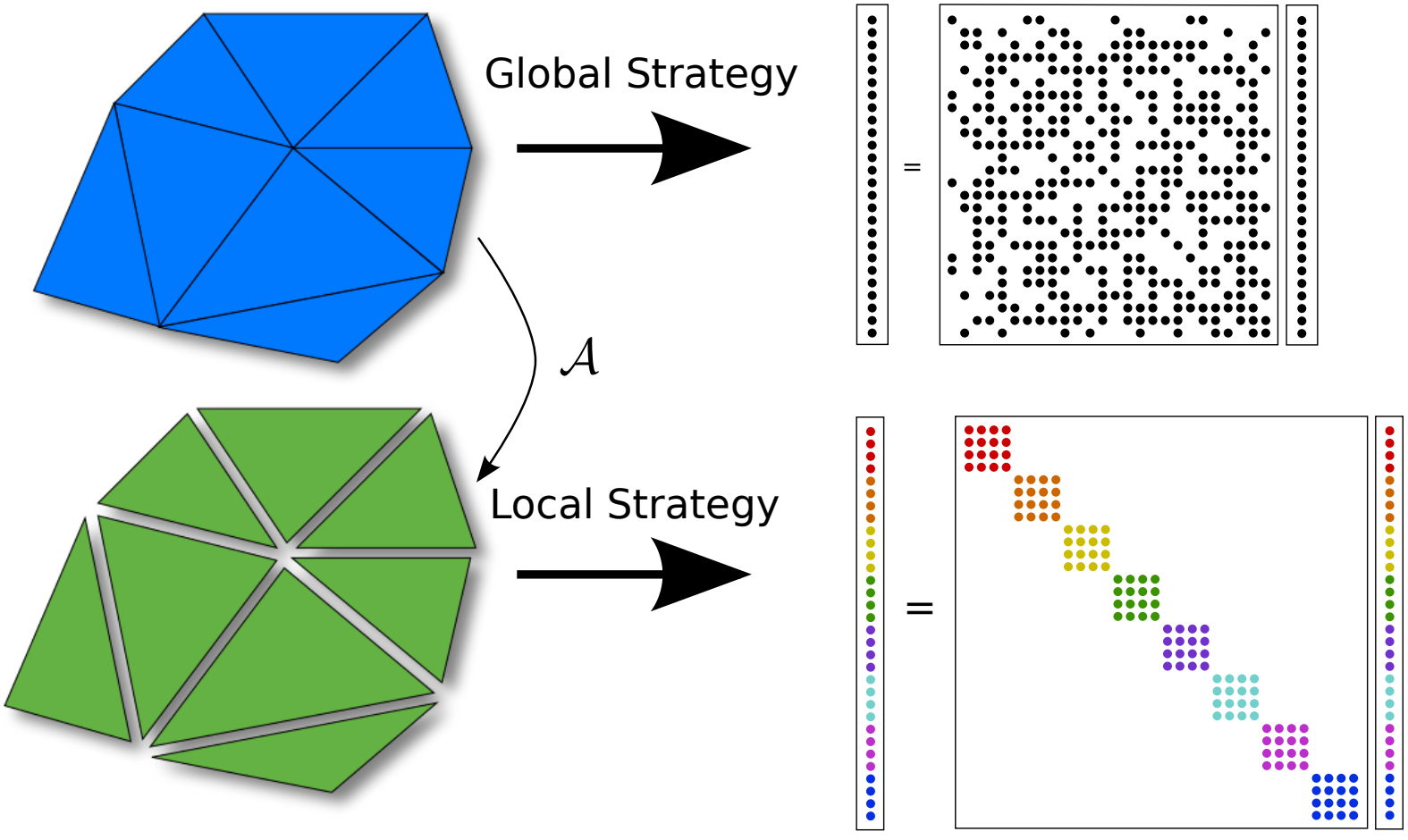



$$= \sum_e \sum_p \int_{\Omega^e} \phi_p(\chi^e(\xi)) f(\chi^e(\xi)) J^e d\xi$$

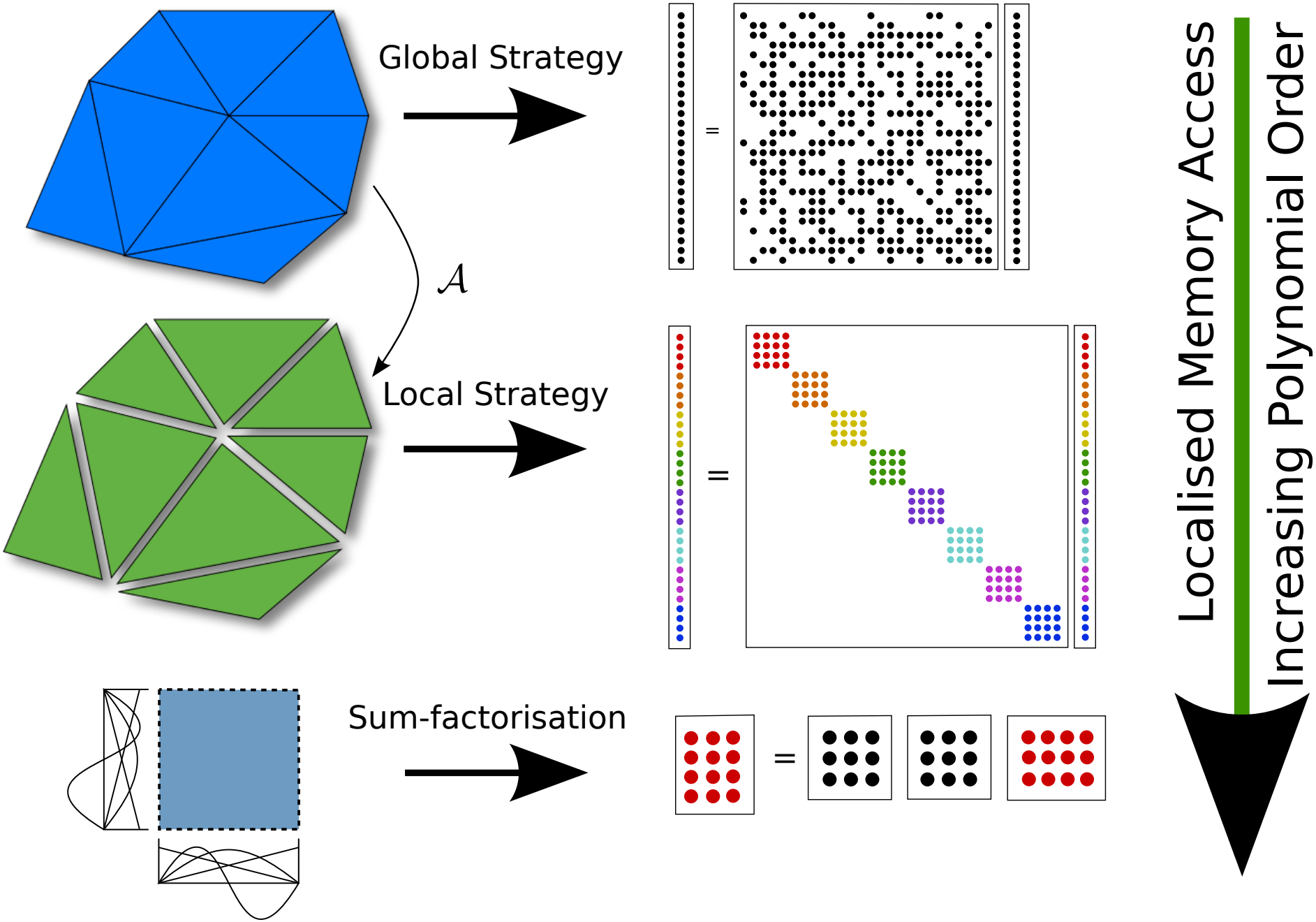
# Implementation strategies



# Implementation strategies

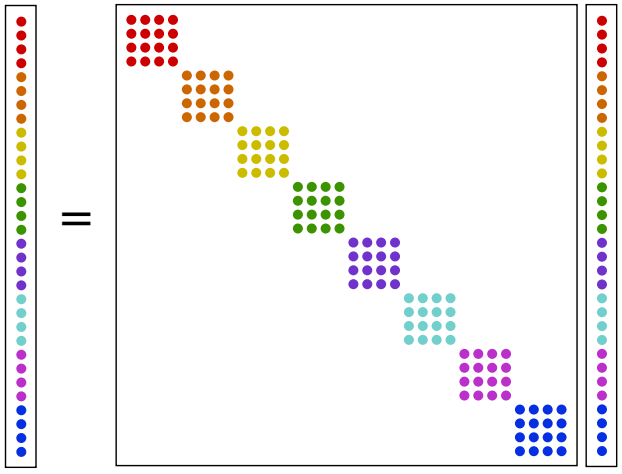


# Implementation strategies



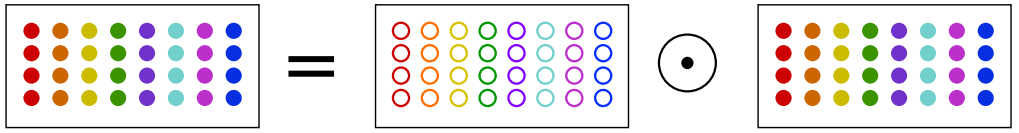
# Collections: Design

## Local Matrix

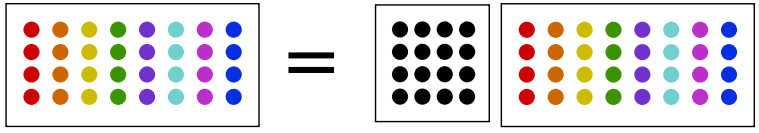


## StdMat

1. Apply Jacobian (L1)

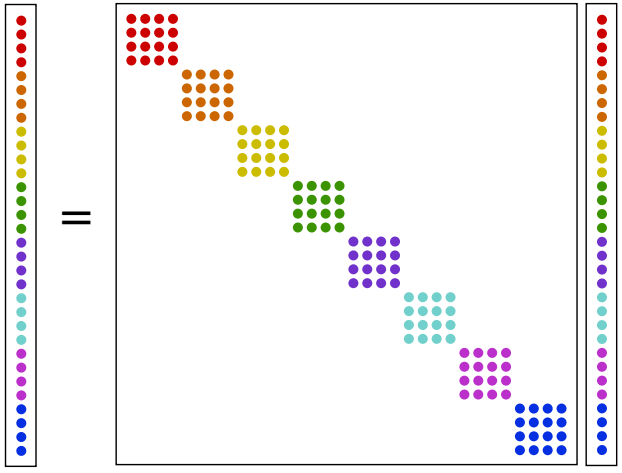


2. L3 Multiply by ref. matrix



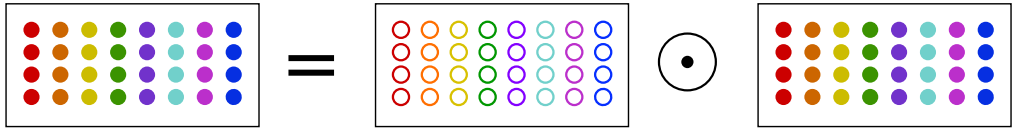
# Collections: Design

## Local Matrix

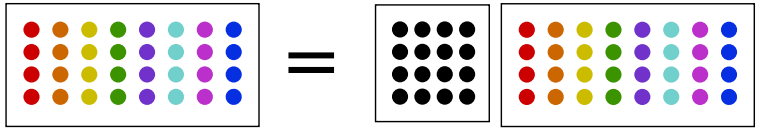


## StdMat

1. Apply Jacobian (L1)



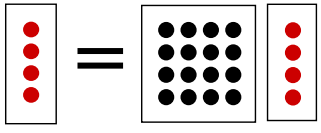
2. L3 Multiply by ref. matrix



## IterPerExp

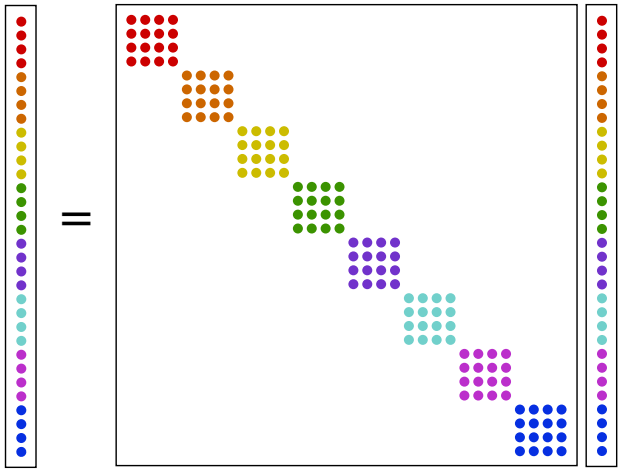
1. Apply Jacobian (L1)
2. L2 multiply by ref. matrix

for i = 1:N



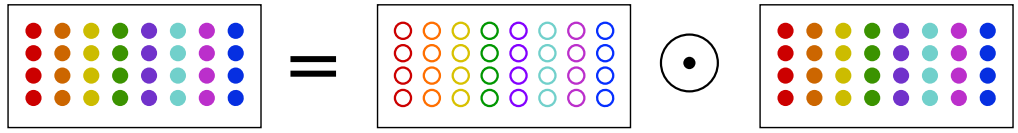
# Collections: Design

## Local Matrix

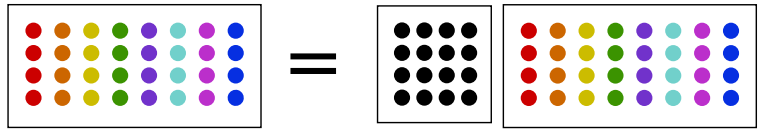


## StdMat

1. Apply Jacobian (L1)



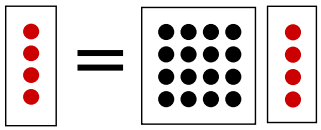
2. L3 Multiply by ref. matrix



## IterPerExp

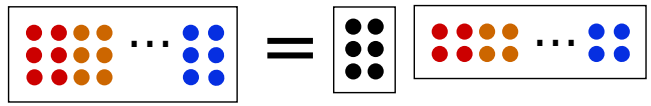
1. Apply Jacobian (L1)
2. L2 multiply by ref. matrix

for i = 1:N



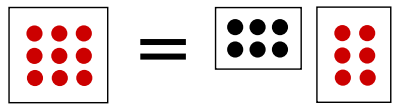
## SumFac (Quad)

1. Apply Jacobian (L1)
2. L2 multiply for first dim ref. matrix



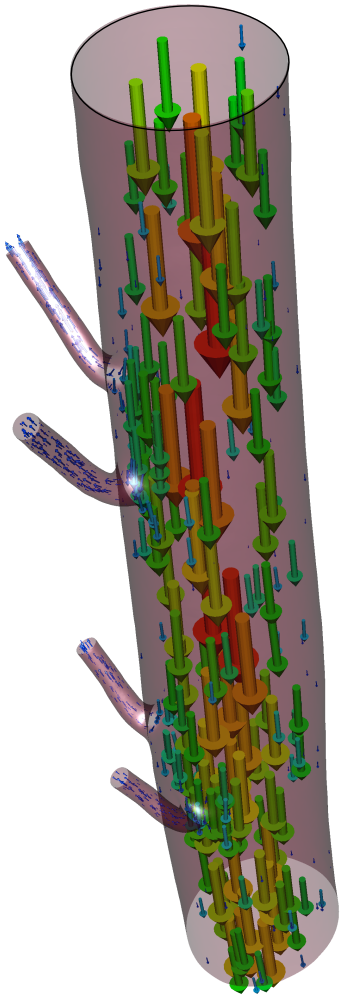
3. L2 multiply for second dim ref. matrix

for i = 1:N



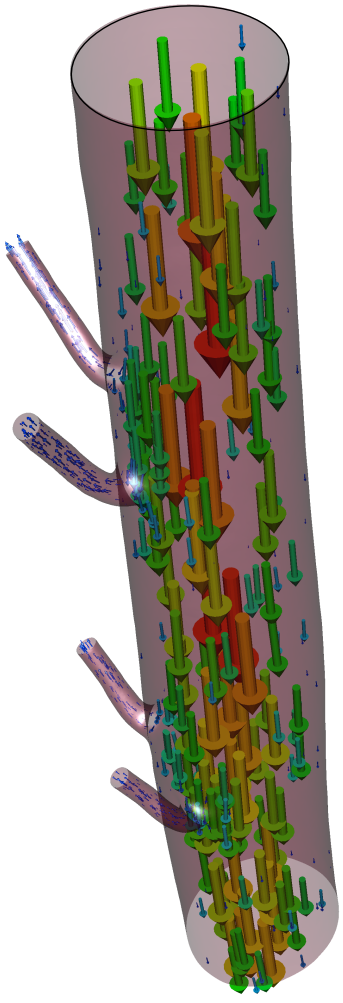
# Test Case

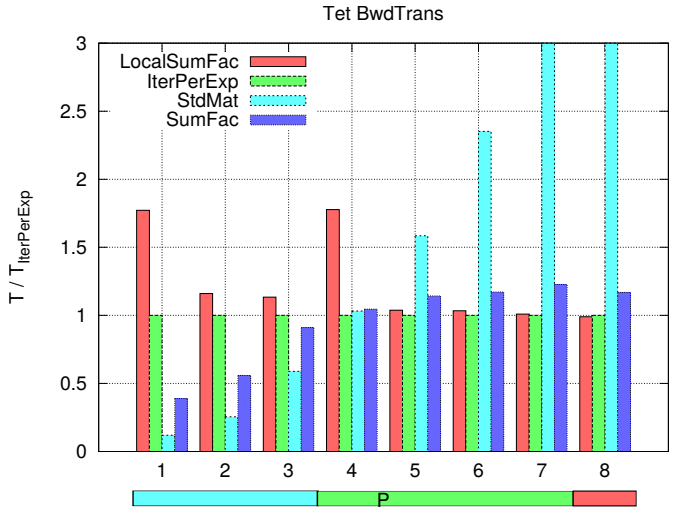
Intercostal Pair
41k tets
21k prisms

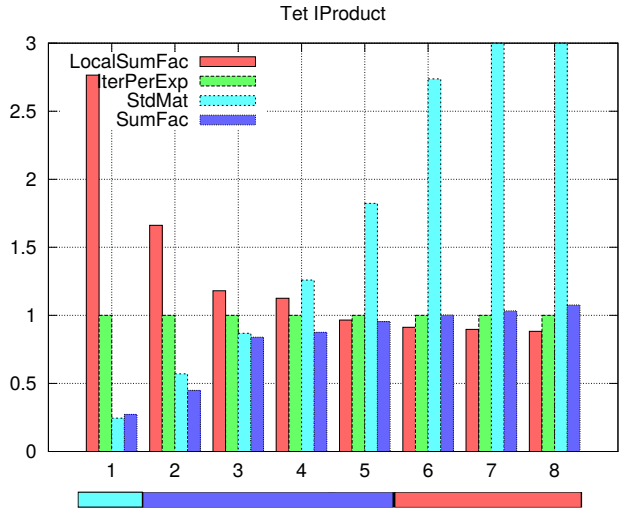


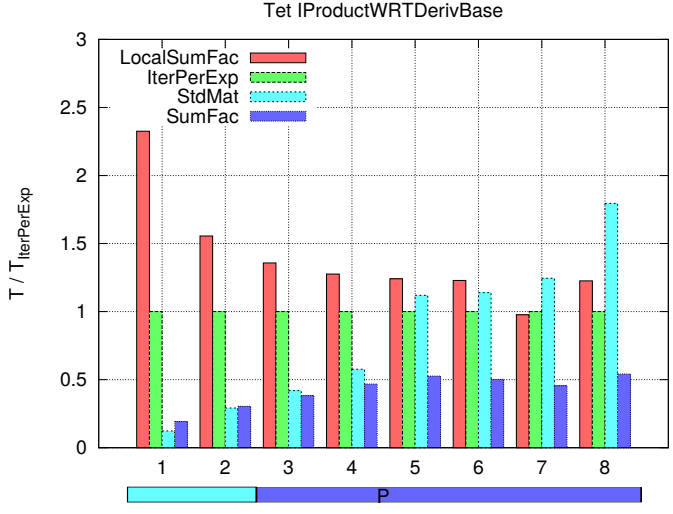
# Test Case

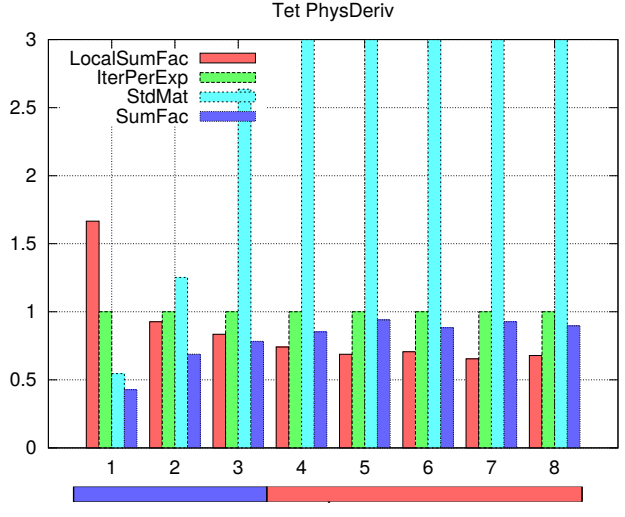
Intercostal Pair
41k tets
21k prisms





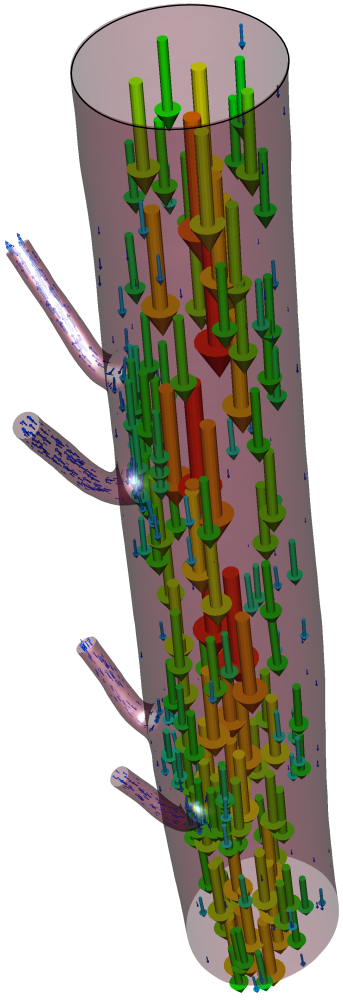



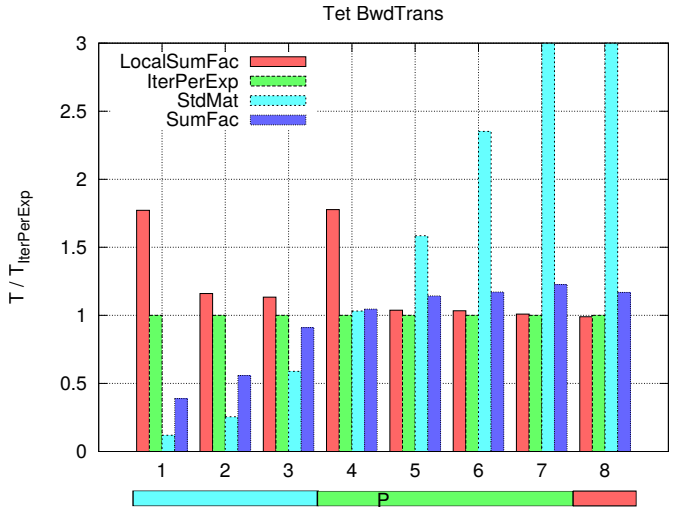



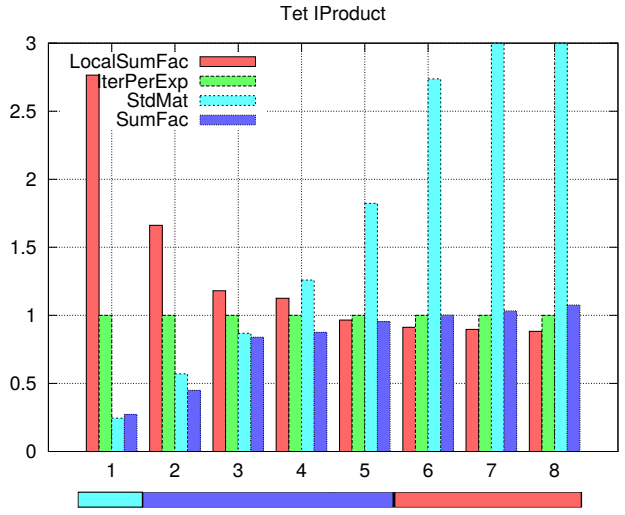



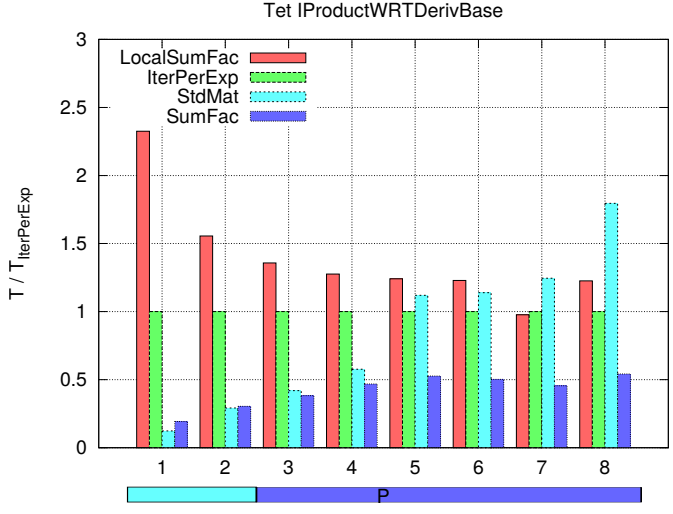
# Test Case

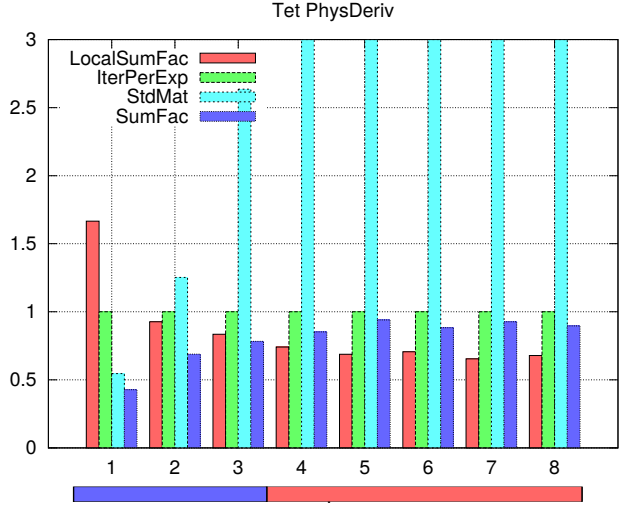
Intercostal Pair
41k tets
21k prisms






StdMat most effective at low order

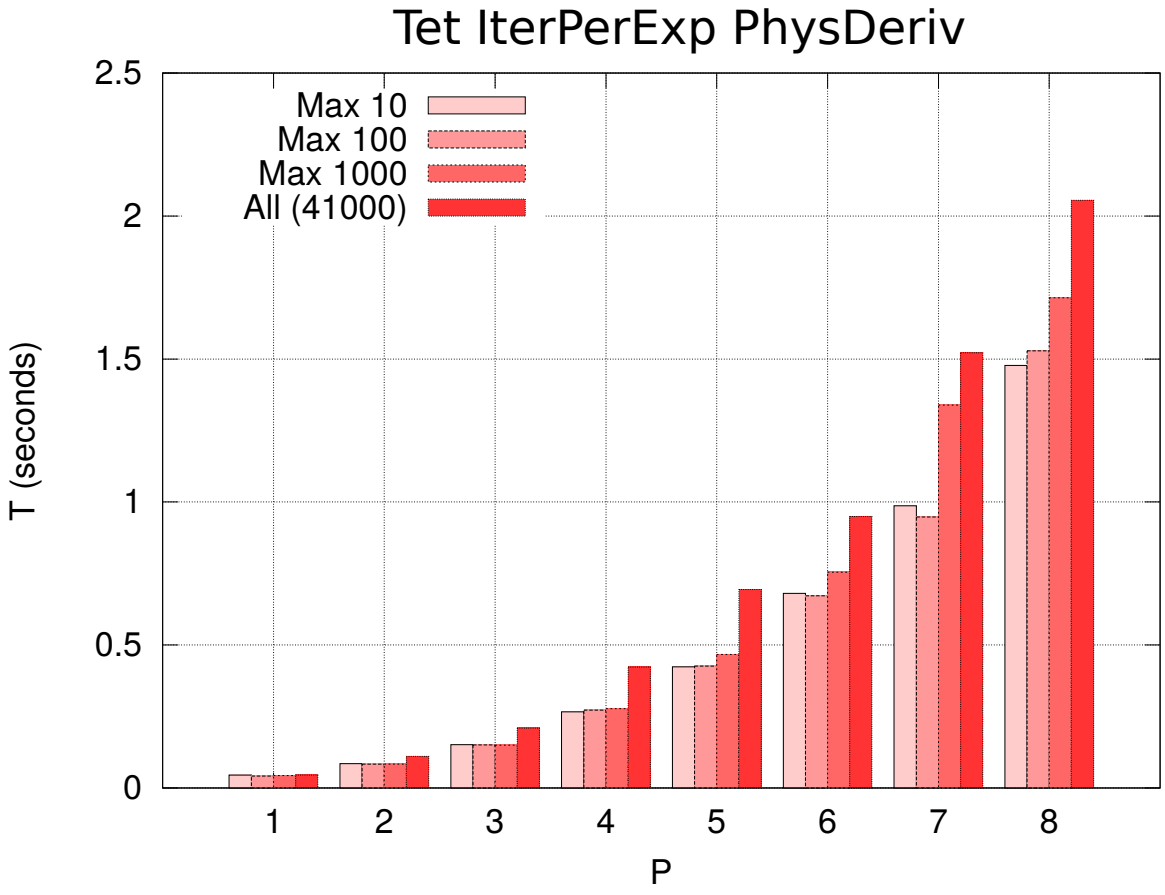
Collections less effective at high order for most operators

PhysDeriv benefits from SumFac even at low P

Similar trends observed for Prisms

# Collection Size Dependence

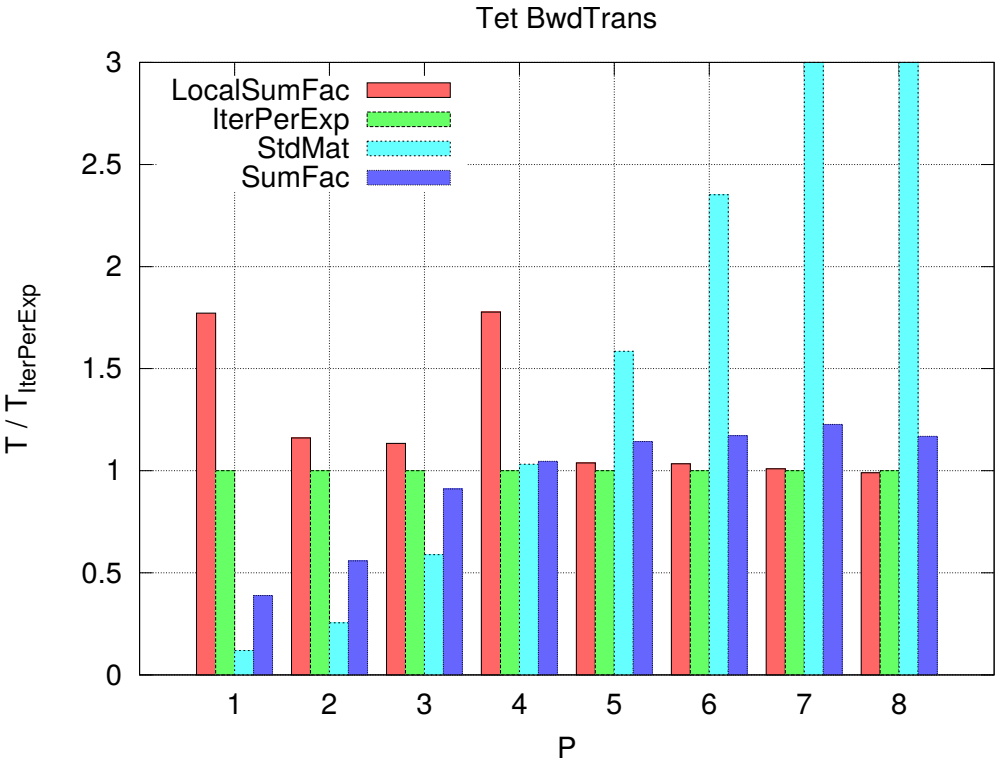
Limit size of collections to 10, 100 or 1000 elements

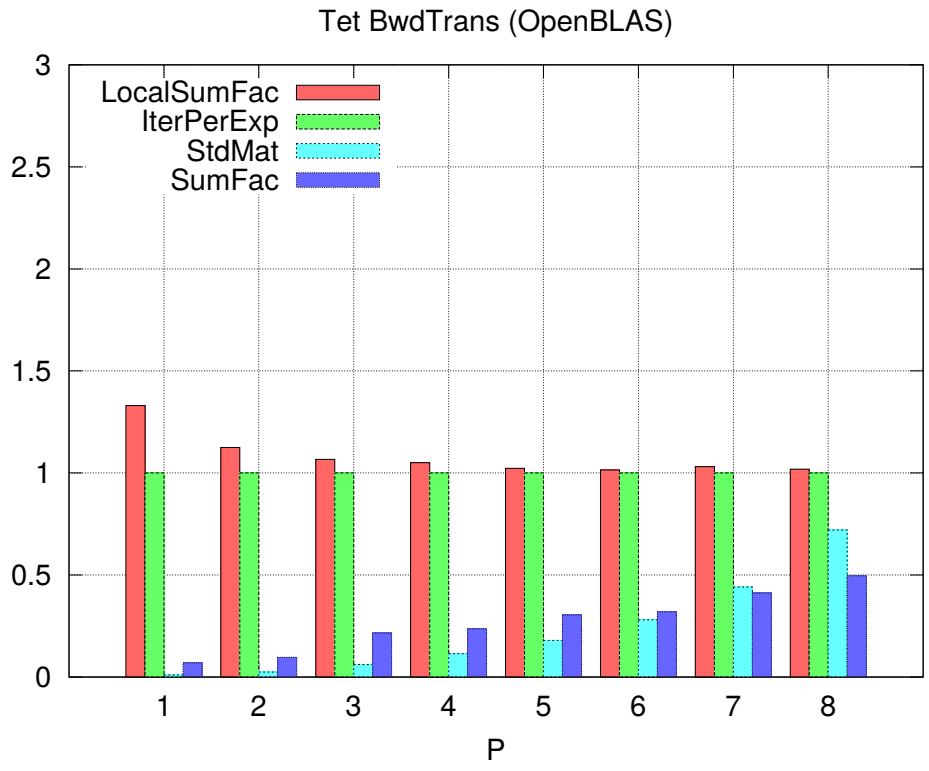



Negligible dependence on collection size for < 1000 elements

Reduced performance for large numbers of elements

# BLAS Implementation




OpenBLAS performs best for large matrix operations

Significant gains for StdMat and SumFac at high P

Importance of auto-tuning

# Developer Practice

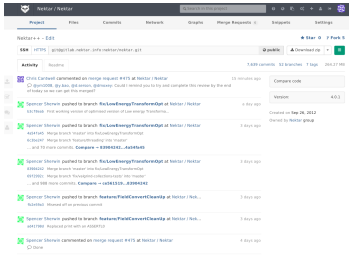
What development practices support a large multi-plaform collaborative software project such as Nektar++?

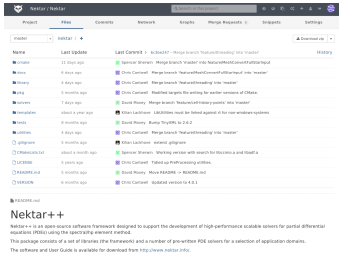
# Developer Practice

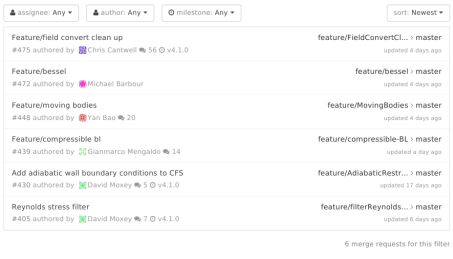
What development practices support a large multi-plaform collaborative software project such as Nektar++?

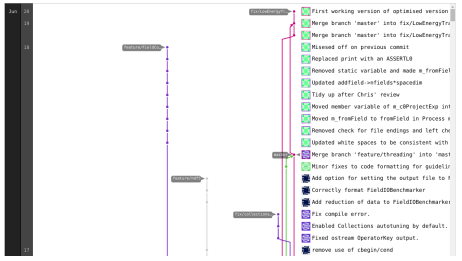
- Version-control (Git + GitLab) http://gitlab.nektar.info

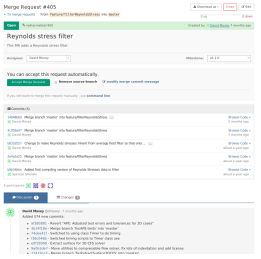
```
git clone https://gitlab.nektar.info/nektar/nektar.git
```



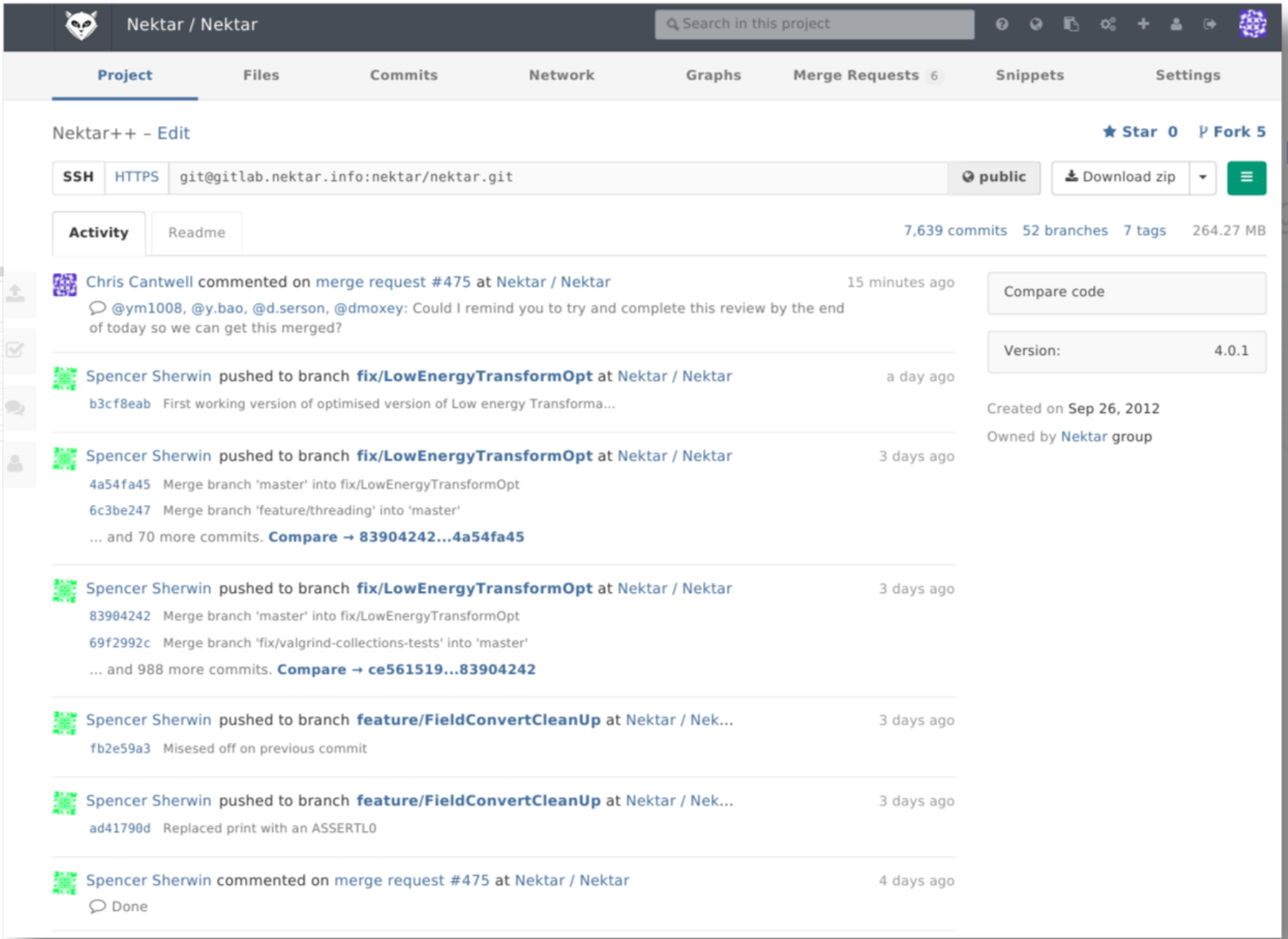


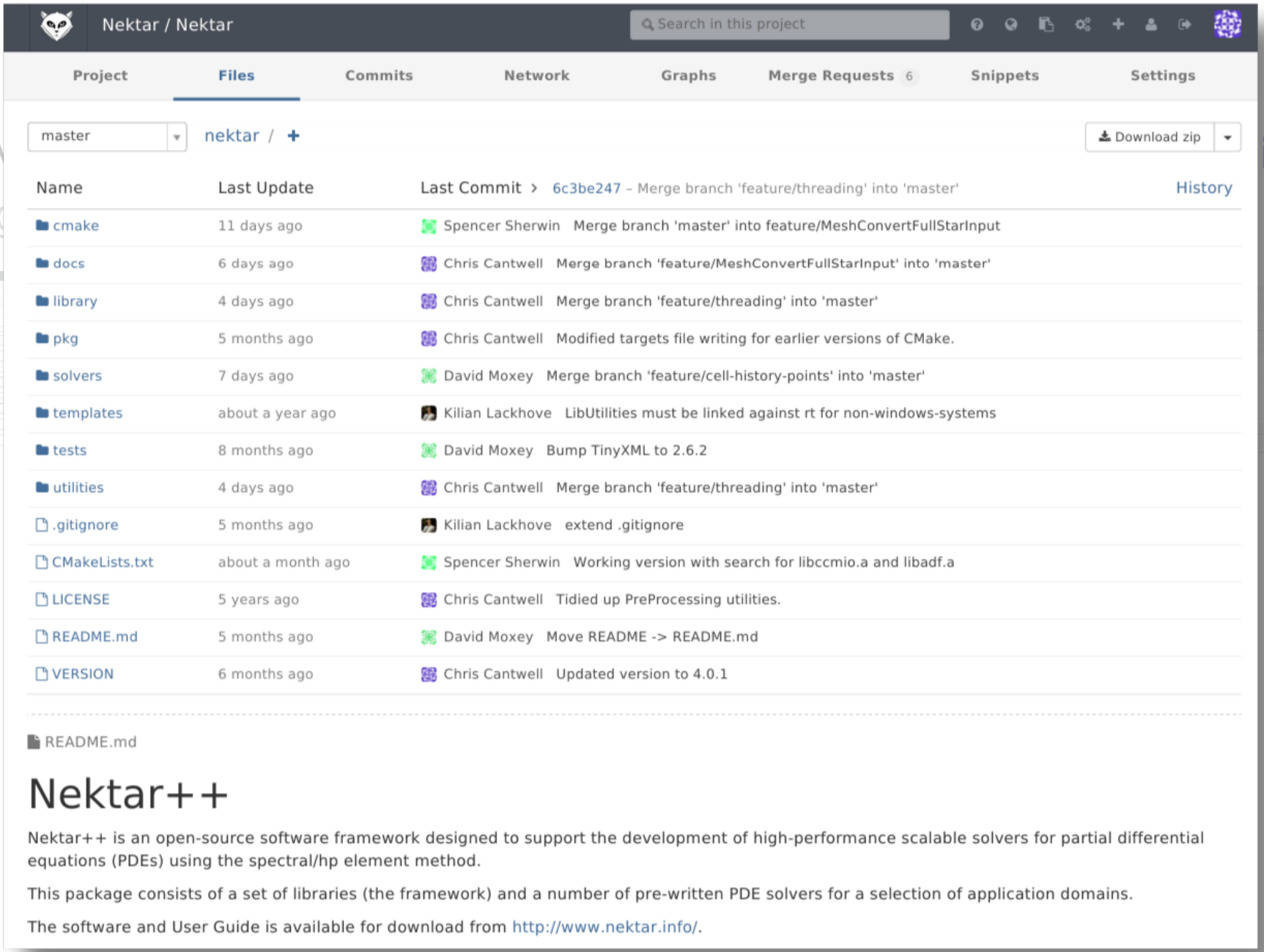





# Developer Practice



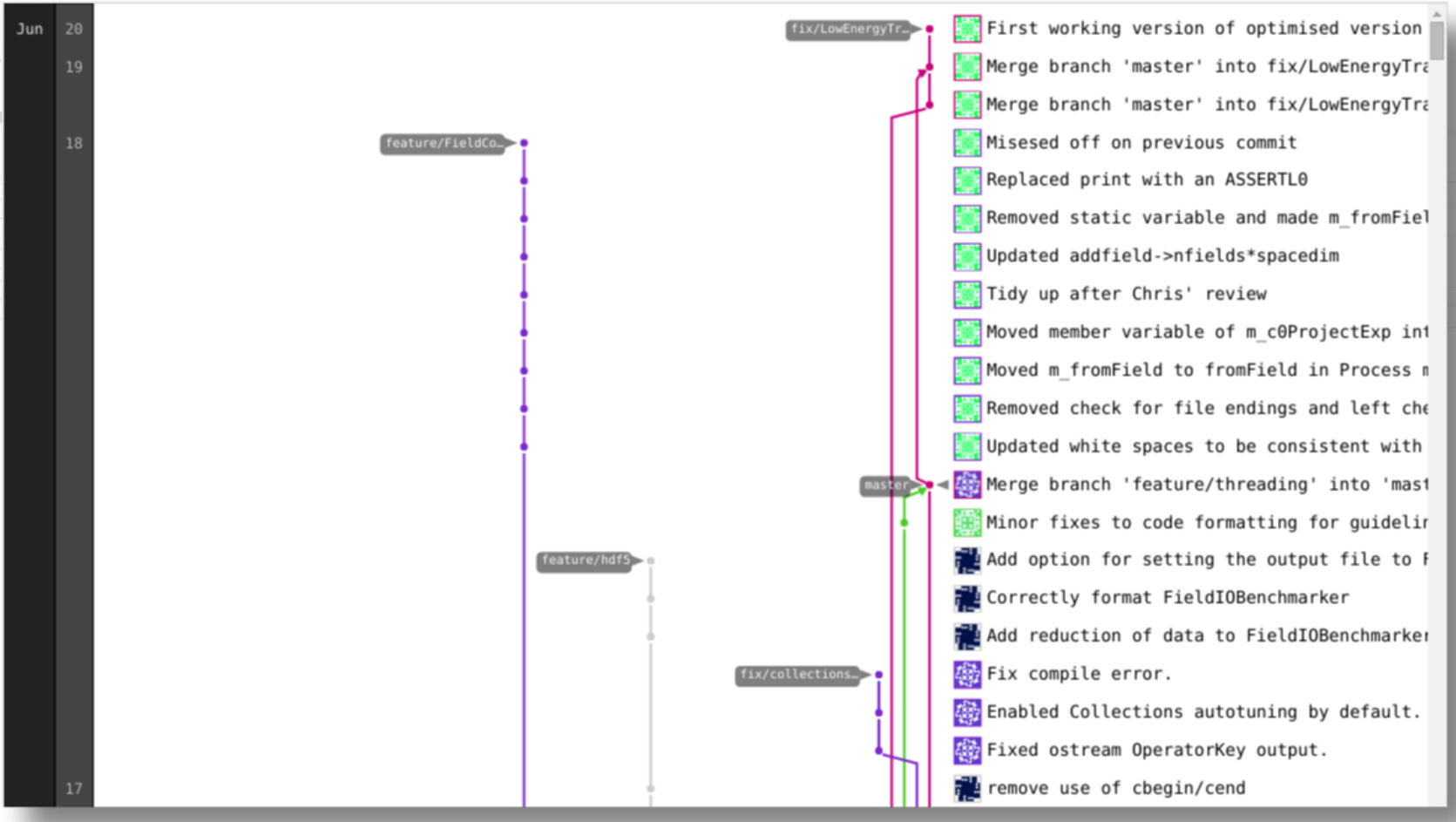
# Developer Practice



# Developer Practice

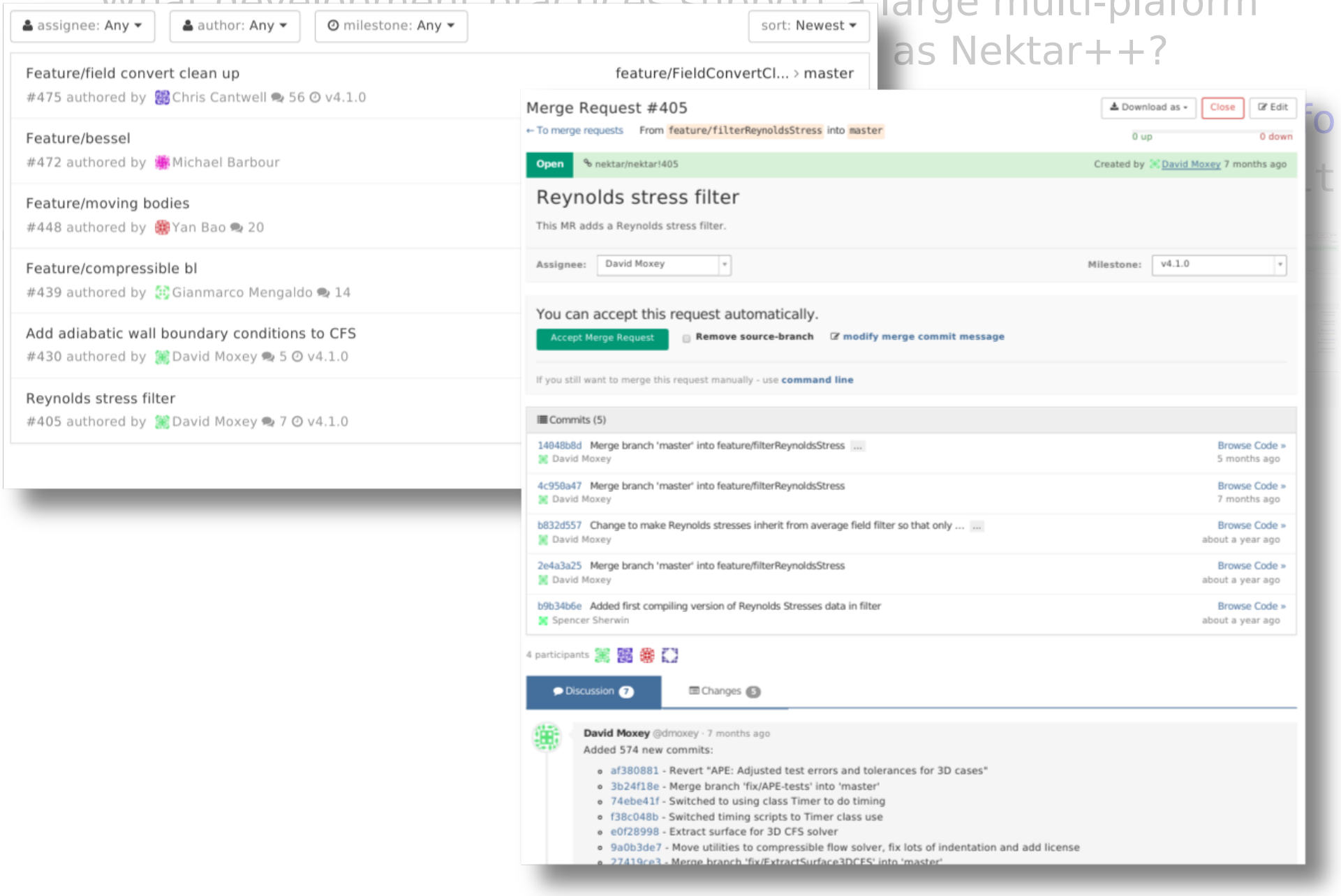
What development practices support a large multi-plaform collaborative software project such as Nektar++?

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



# Developer Practice

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

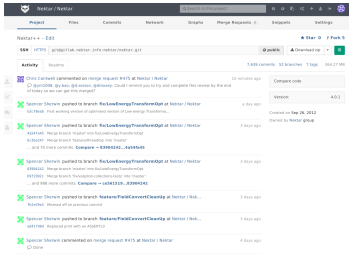
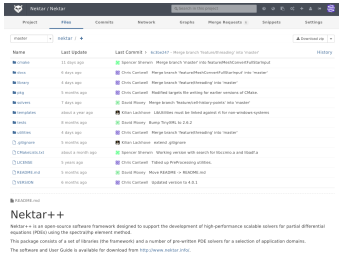
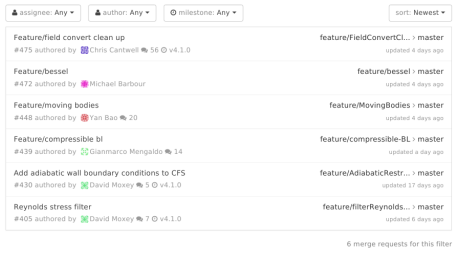
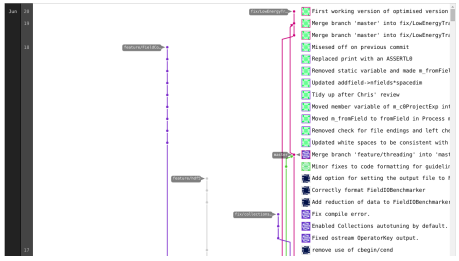
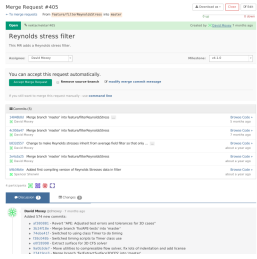


# Developer Practice

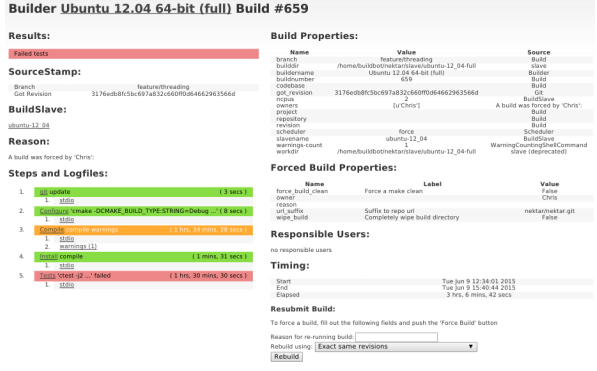
What development practices support a large multi-plaform collaborative software project such as Nektar++?

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

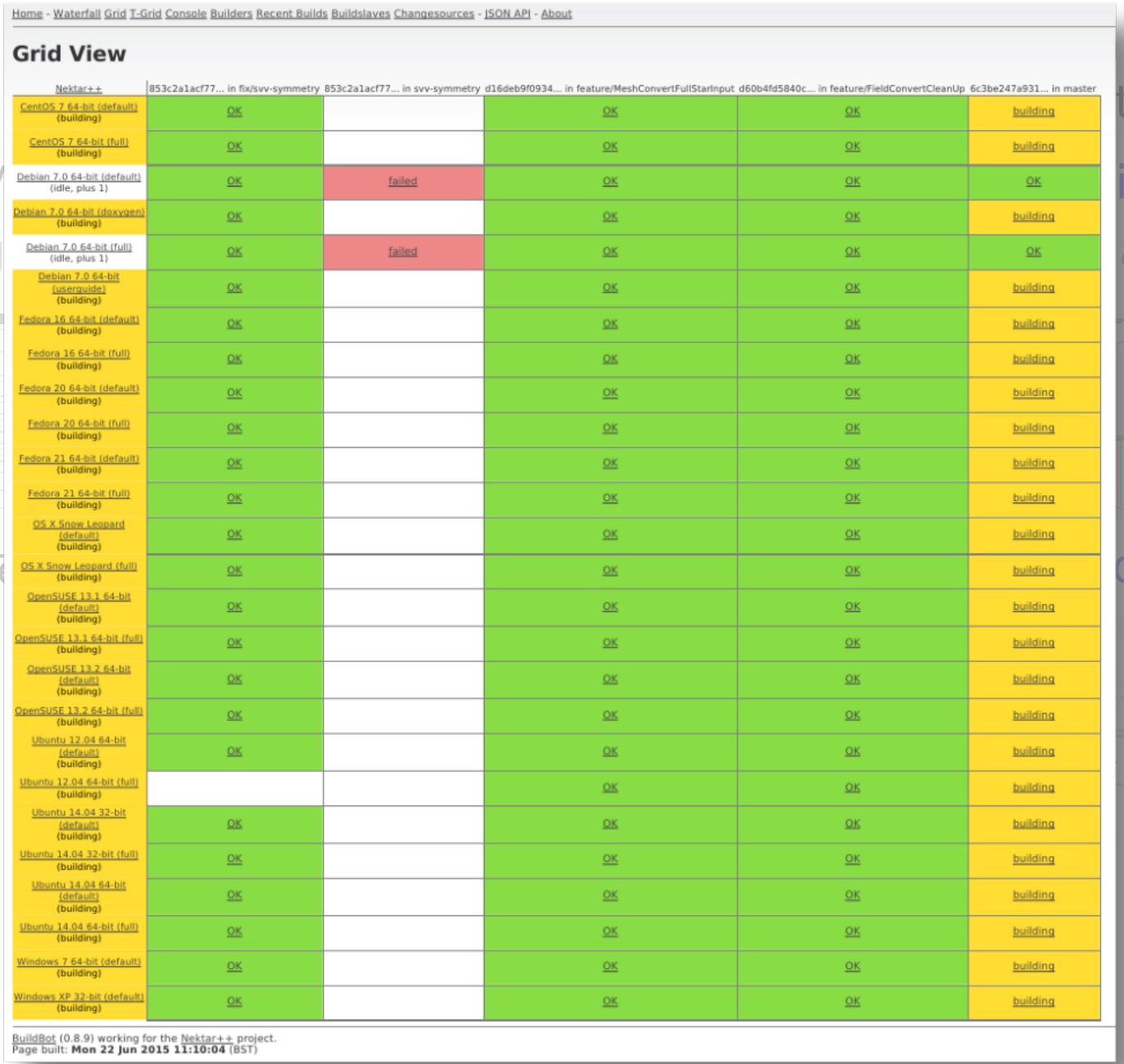
```
git clone https://gitlab.nektar.info/nektar/nektar.git
```

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

# Developer Practice



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

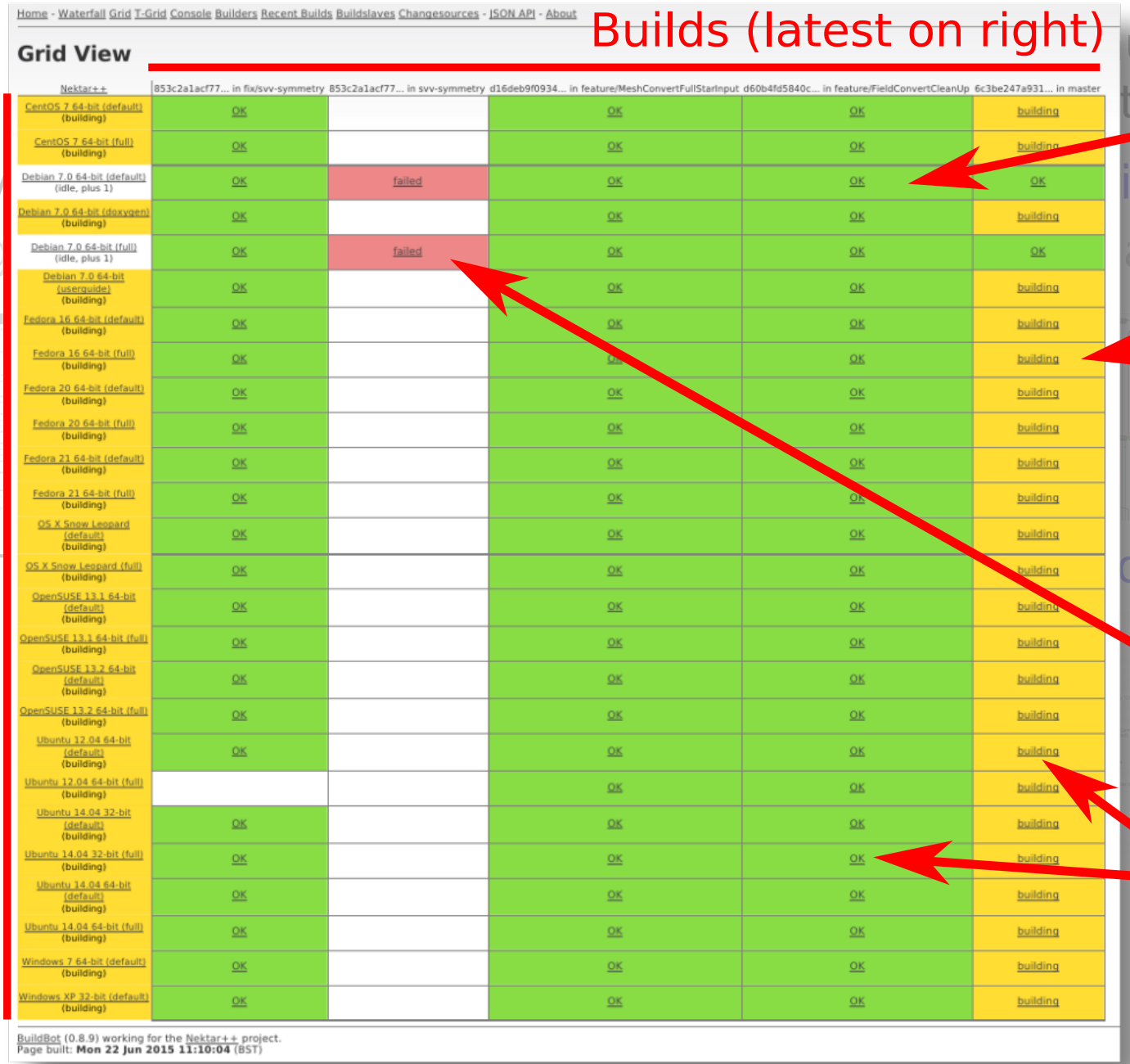
tar++?

itlab.nektar.info

ar/nektar.git

dbot.nektar.info

# Developer Practice



# Developer Practice

## 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

# Developer Practice

## 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)

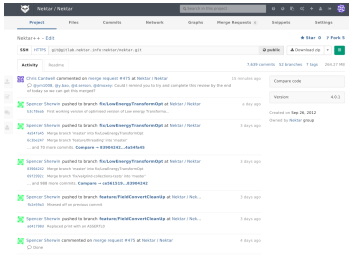
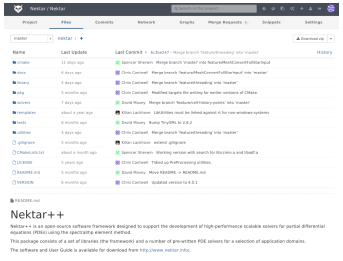
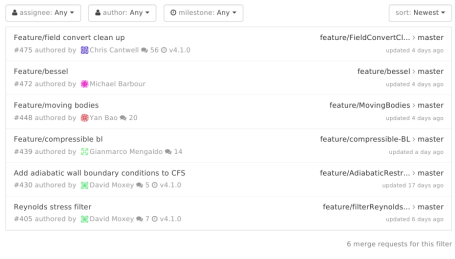
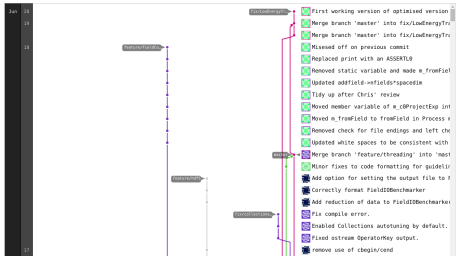
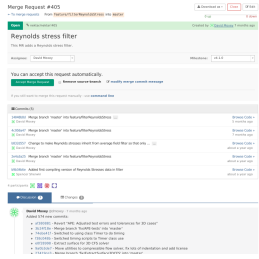
```
ctest -j2 '--timeout 600' --output-on-failure
 in dir /home/buildbot/nektar/slave/ubuntu-12_04-full/build/builds (timeout 1200 secs)
 watching logfiles {}
 argv: ['ctest', '-j2', '--timeout 600', '--output-on-failure']
 environment:
  HOME=/home/buildbot
  LANG=en_GB.UTF-8
  LANGUAGE=en_GB:en
  LOGNAME=buildbot
  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
```

# Developer Practice

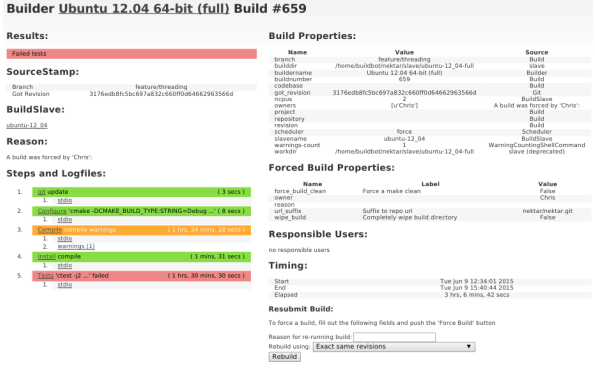
What development practices support a large multi-plaform collaborative software project such as Nektar++?

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

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

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

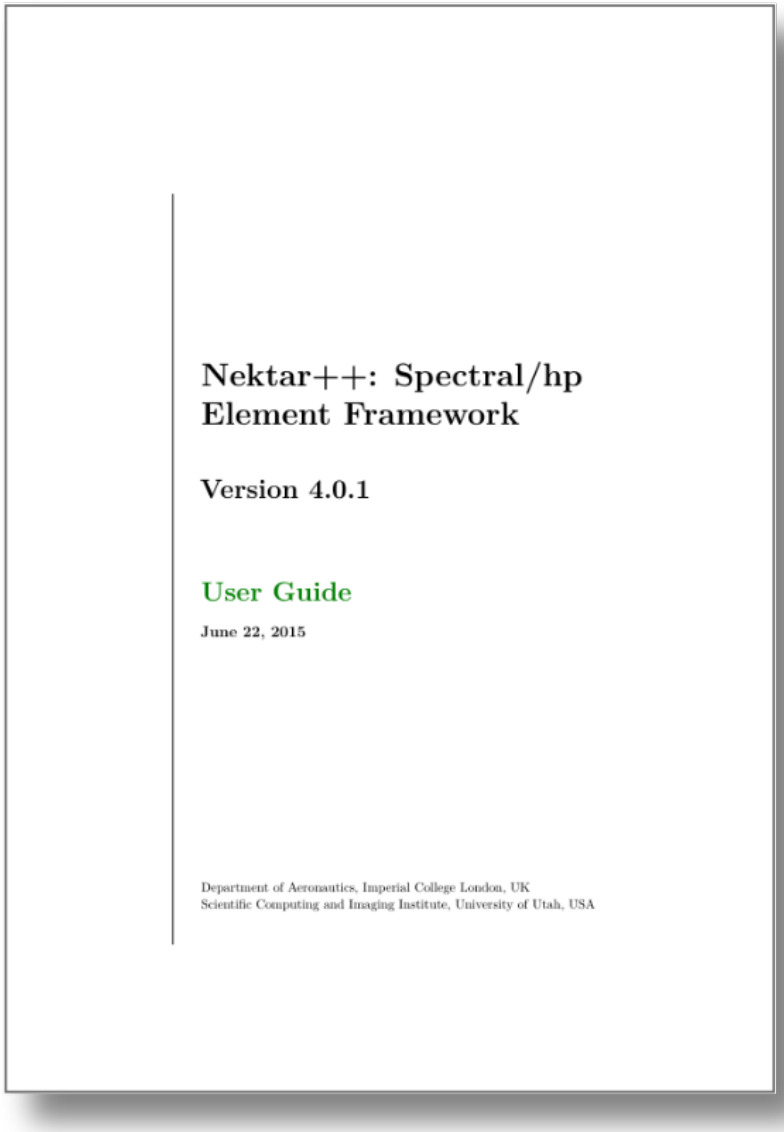

- Issue tracking (Trac) http://www.nektar.info
- Documentation (PDF + doxygen) http://doc.nektar.info

# Documentation

- All documentation versioned with code

# Documentation

- All documentation versioned with code


Nektar++: Spectral/hp
Element Framework

Version 4.0.1

User Guide

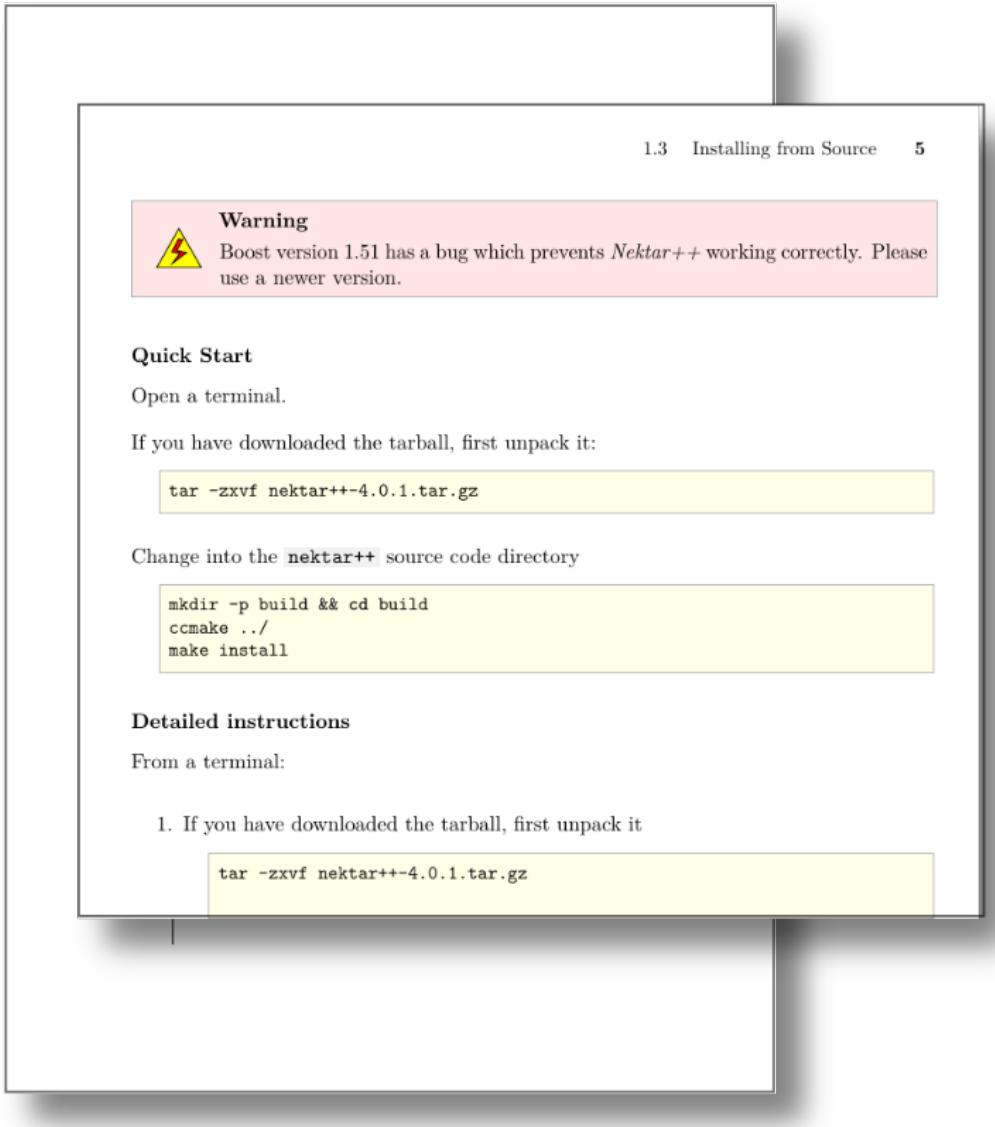
June 22, 2015

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

- User guide

# Documentation

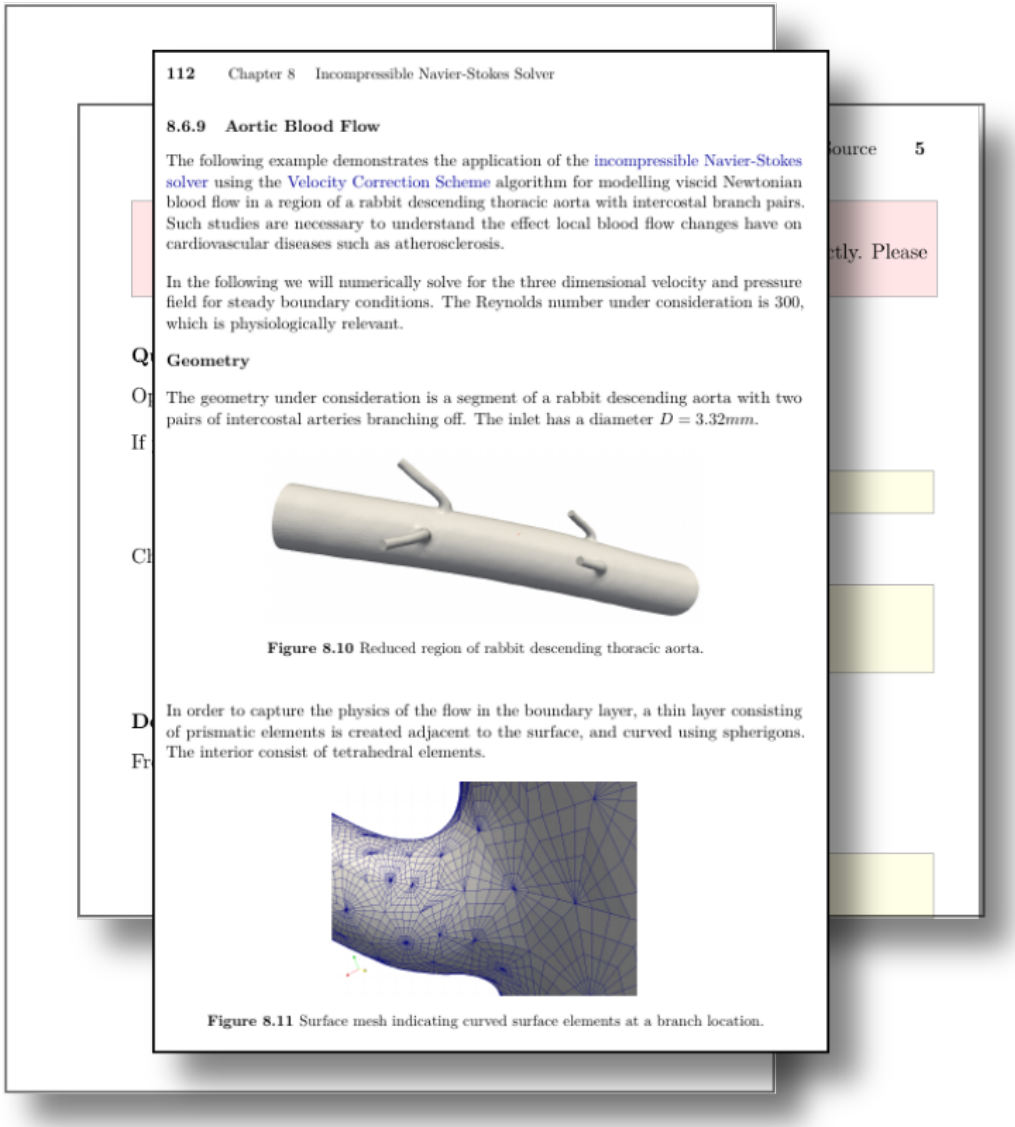
- All documentation versioned with code




- User guide
  - Installation

# Documentation

- All documentation versioned with code



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.

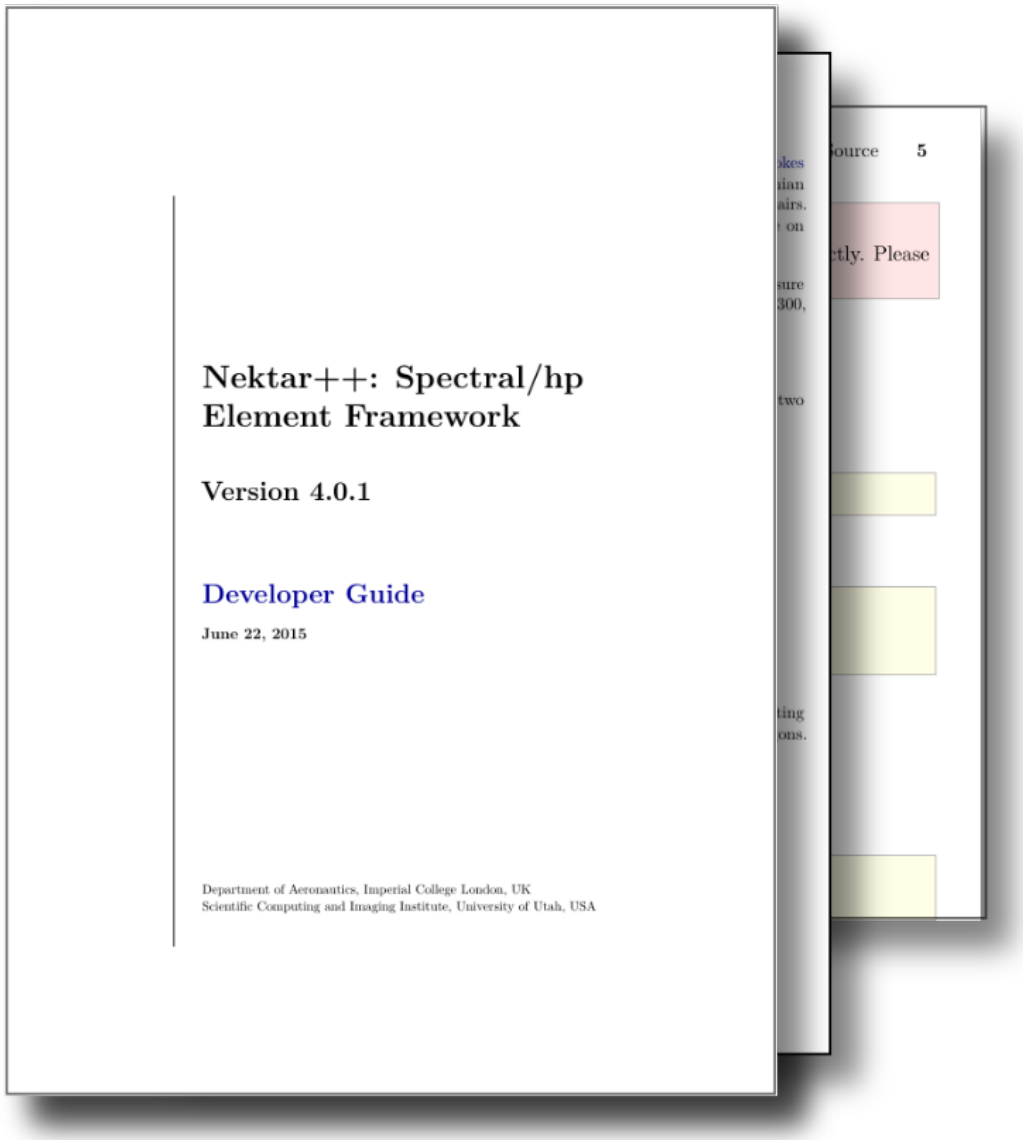
In order to capture the physics of the flow in the boundary layer, a thin layer consisting of prismatic elements is created adjacent to the surface, and curved using spherigons. The interior consist of tetrahedral elements.

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

- User guide
  - Installation
  - Solver and Utility usage
  - Tutorials
  - Input file reference
  - FAQs

# Documentation

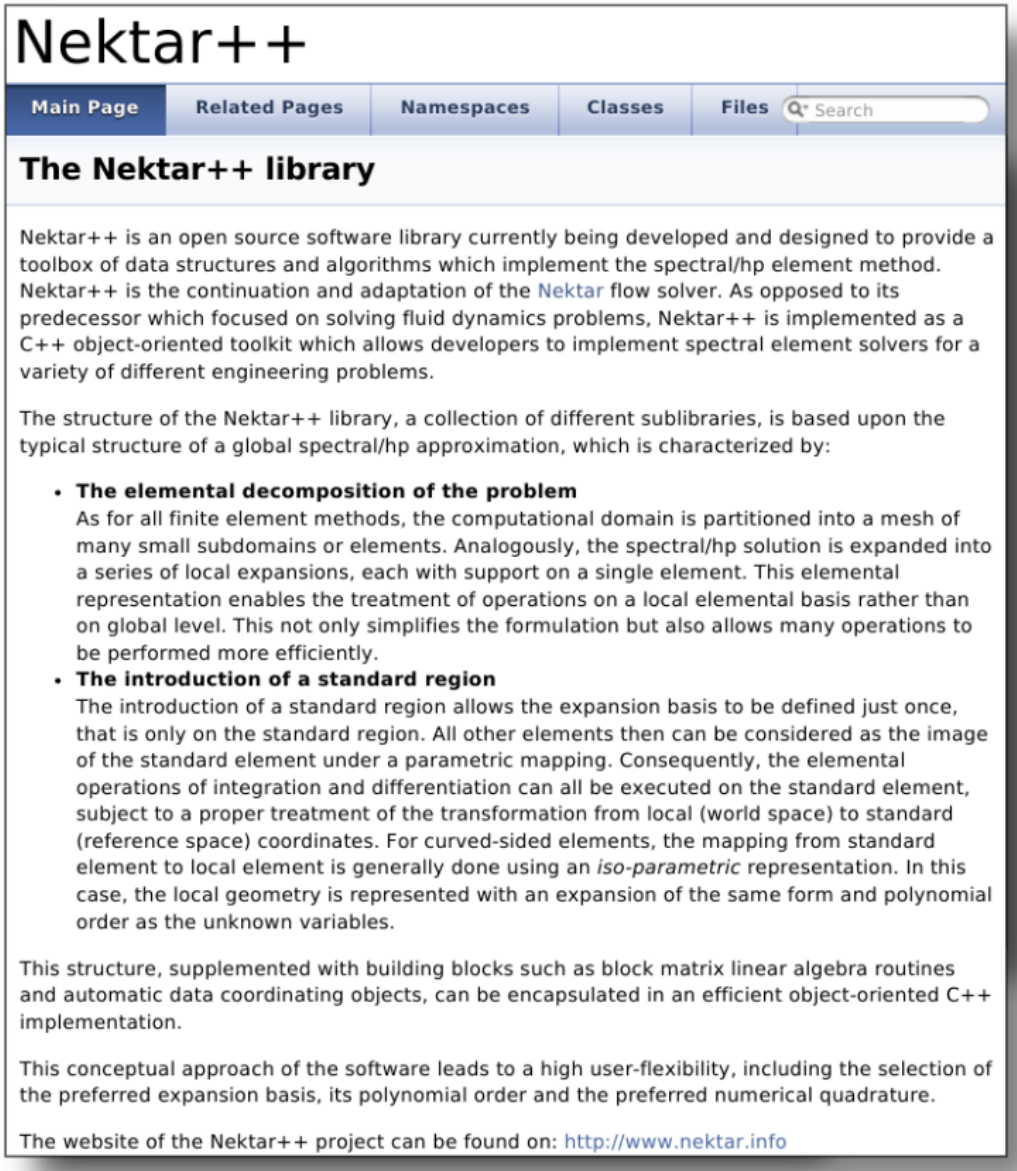
- All documentation versioned with code




- User guide
  - Installation
  - Solver and Utility usage
  - Tutorials
  - Input file reference
  - FAQs
- Developer guide
  - Programming concepts
  - Library design
  - Nektar++ structures
  - Nektar++ algorithms
  - Coding Standard

# Documentation

- All documentation versioned with code



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

- User guide
  - Installation
  - Solver and Utility usage
  - Tutorials
  - Input file reference
  - FAQs
- Developer guide
  - Programming concepts
  - Library design
  - Nektar++ structures
  - Nektar++ algorithms
  - Coding Standard
- Code documentation
  - Doxygen
  - Detailed implementation specifics

# Summary

- Design of the Nektar++ framework
  - Stack of libraries
  - Code design mirrors mathematical formulation
  - **Encapsulate** complexities of spectral/$hp$ element methods
  - Improve **accessibility**
- Collections
  - Multiple implementations for **achieving performance across P**
  - Collates action of operators across **multiple elements**
  - Improves **efficiency** on modern vector-capable CPUs
  - Easy to use with **auto-tuning**
- Developer practice
  - Version control (Git + Gitlab)
  - Regression tests (also useful as examples!)
  - Continuous integration (Buildbot)
  - Issue tracking (Trac)
  - Documentation (LaTeX + Doxygen)

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, in press, 2015