Science & Technology
Facilities Council

Software Engineering Requirements for the CCP-WSI code repository

Gemma Poulter

gemma.poulter@stfc.ac.uk

The CCP-WSI Code Repository

- A major goal of CCP-WSI:
  - Bring the community together
  - Place code development on firm and sustainable footing
  - Computational WSI facility for all groups

- The CCP-WSI code repository has been setup and welcomes contributions!
  - Hosted by CCPForge
  - Uses Git version control software
  - Rules, guidelines and advice on how to contribute all available on project wiki.

The CCP-WSI Code Repository

- A major goal of CCP-WSI:

Project wiki:
http://www.softeng.rl.ac.uk/wiki/CCP-WSI

To register:
http://www.softeng.rl.ac.uk/wiki/CCP-WSI/Admin?action=newaccount

welcomes contributions!

- Hosted by CCPFor
- Uses Git version control software
- Rules, guidelines and advice on how to contribute all available on project wiki.

Science & Technology Facilities Council

The CCP-WSI Code Repository

To gain access to the repository:

1. Register for CCPForge: https://ccpforge.cse.rl.ac.uk/gf/

2. Request to join the CCP-WSI project: https://ccpforge.cse.rl.ac.uk/gf/project/ccpwsi/

3. Once approved, clone the repository:

git clone ssh://<username>@ccpforge.cse.rl.ac.uk/gitroot/ccpwsi

Science & Technology Facilities Council

The CCP-WSI Code Repository

```
└── ccp-wsi
    ├── OF3.0.1
    │   ├── README
    │   ├── libraries
    │   │   └── README
    │   ├── models
    │   │   └── README
    │   ├── platforms      --Note this is for binaries which should not be committed to Git
    │   │   └── README
    │   ├── solvers
    │   │   └── README
    │   ├── test_cases
    │   │   └── README
    │   ├── tutorials
    │   │   └── README
    │   ├── utilites
    │   │   └── README
    │   ├── Allwclean
    │   └── Allwmake
    └── OF4.0
Additional versions of OpenFoam or FoamExtend will appear here
```

Science & Technology Facilities Council

Software Engineering Requirements for the CCP-WSI Code Repository

- Version Control (Git) and development process
- Coding Standards
- Documentation
- Continuous Integration (SESC build service)

Science & Technology Facilities Council

Software Engineering Requirements for the CCP-WSI Code Repository

- **Version Control (Git) and development process**
- Coding Standards
- Documentation
- Continuous Integration (SESC build service)

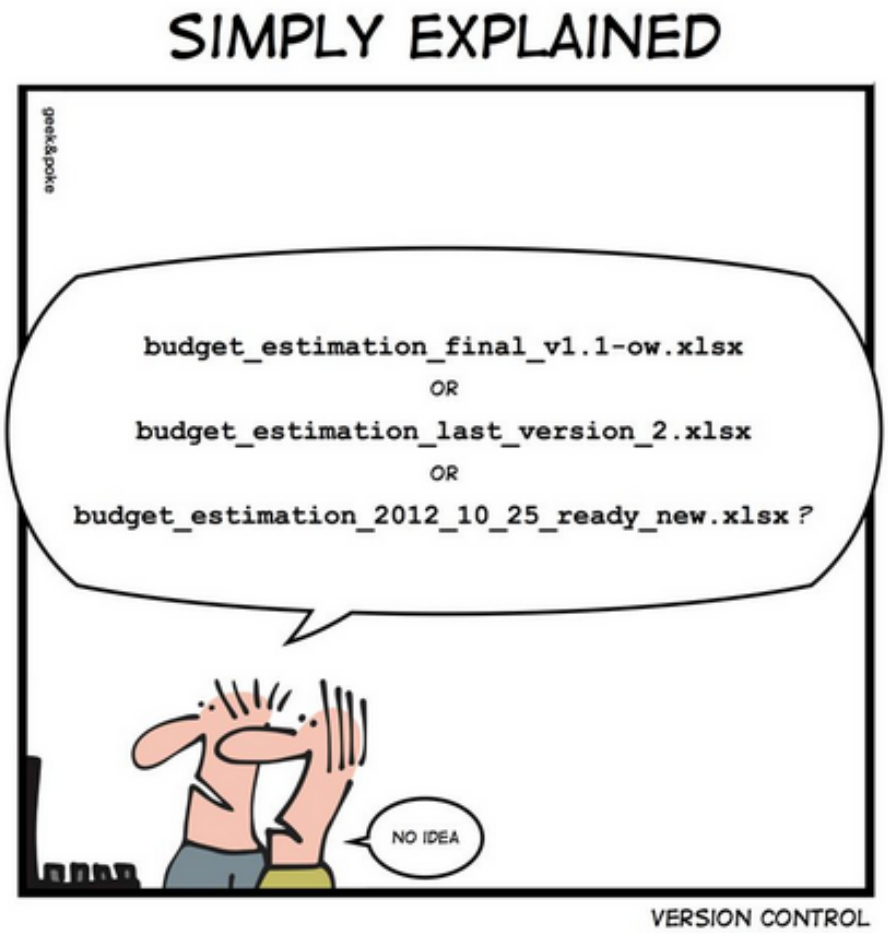
Science & Technology Facilities Council

Version Control Systems

“Version control is a system that records changes to a file or set of files over time so that you can recall specific versions later.” – Pro Git Book

Version Control Systems

- No longer need to copy files around to keep previous versions
  - No more mazes of folders
  - No more copying files to USB drives
- Find out why changes were made, who made then and when they happened
- Makes collaboration easier
  - No fear about overwriting work
  - No copying to network drives / emailing files around
  - Support for better workflows



Git Overview

- The CCP-WSI code repository uses Git
- Files can be in one of four states
  - Untracked – not managed by Git
  - Unmodified – managed by Git, no changes
  - Modified – managed by Git, has changes since the last version
  - Staged – managed by Git, has changes which are marked to be part of the next commit
- Once you are happy with the staged changes you commit them, adding a descriptive message

Basic git operations: Pushing and pulling

- To download updates from CCPForge

  git pull

- To upload changes to CCPForge

  git push

Basic git operations: Useful commands

- View current status of working copy and index

  git status

- View changes

  git diff

- View commit history

  git log

Basic git operations: Adding or editing a file

- Add the change to the index (staging area)

  git add *<file>*

- Commit the changes in the index to the local repository

  git commit

- Upload the changes in the local repository to CCPForge

  git push

Basic git operations: Deleting a file

- Delete the file and remove it from the index (staging area)

  git rm *<file>*

- Commit the changes in the index to the local repository

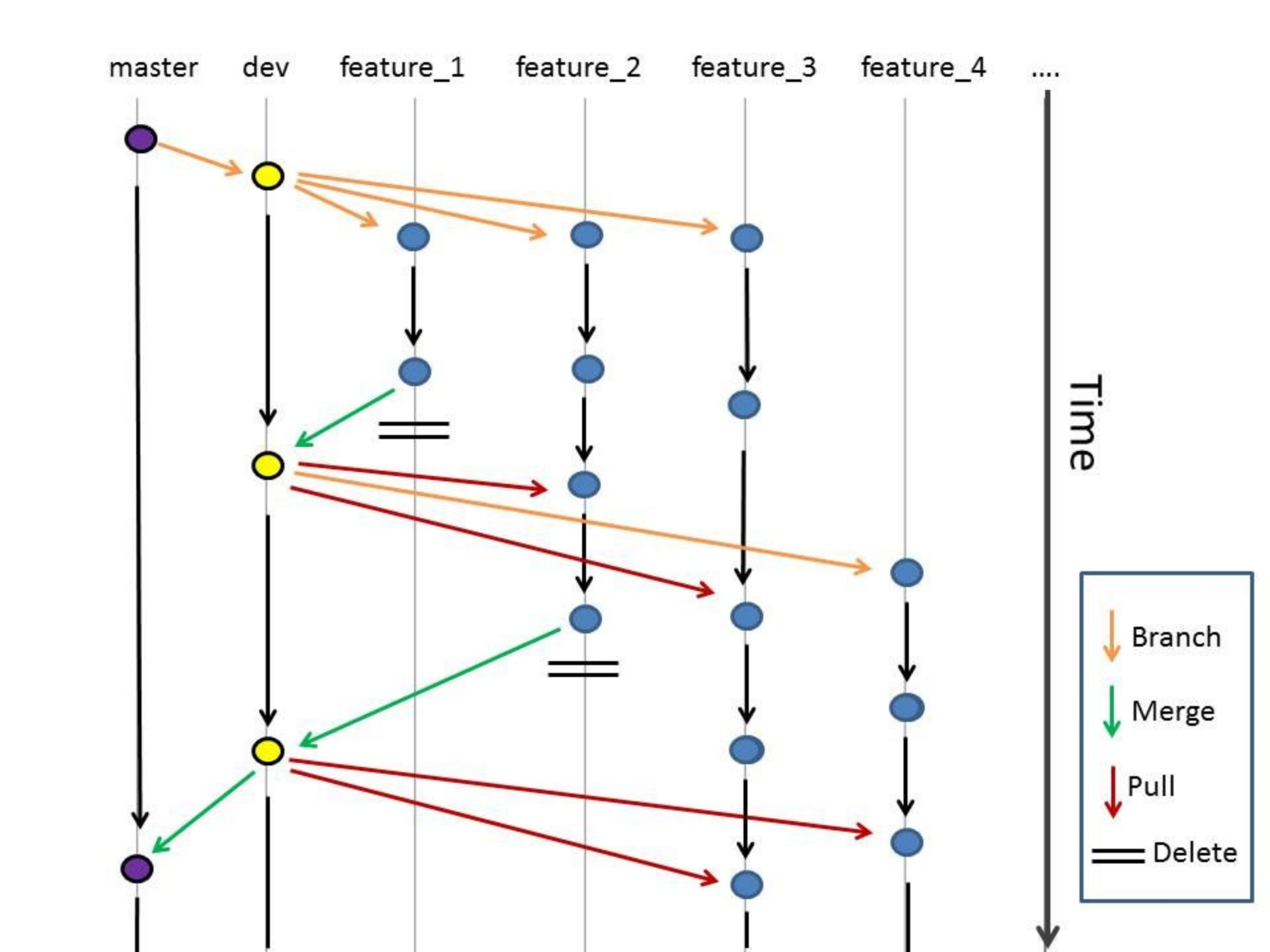
  git commit

- Upload the changes in the local repository to CCPForge

  git push

Branching & Merging

- Branches simplify development
  - Allows splitting from the mainline so as not to break it
  - Useful to separate development of different features and bug fixes
  - Especially useful with many collaborators
- Once work is finished it can be merged back into the mainline again
  - Possibility causing conflicts if others have worked on the same areas of the files
- Contributions to the CCP-WSI code repository are through a branching and merging process:


master
dev
feature_1
feature_2
feature_3
feature_4
....
Time
Branch
Merge
Pull
Delete

Git and development process help

- There is a FAQ page on the wiki:

http://www.softeng.rl.ac.uk/wiki/CCP-WSI/SoftwareDevelopment/GitProblems

- Please ask if anything is unclear.

gemma.poulter@stfc.ac.uk

Software Engineering Requirements for the CCP-WSI Code Repository

- Version Control (Git) and development process
- **Coding Standards**
- Documentation
- Continuous Integration (SESC build service)

Coding Standards

- Currently repository only has code using OpenFOAM
- Please adhere where possible to the OpenFOAM coding standards:

  http://openfoam.org/dev/coding-style-guide/

- Naming conventions for files, directory structure etc are detailed on the project wiki:

http://www.softeng.rl.ac.uk/wiki/CCP-WSI/SoftwareDevelopment/Rules

Documentation

- Currently just “README” files in the repository
- We are working on a more formal and structured documentation format
  - informative and useful to users
  - as un- tedious and laborious as possible to developers

SESC Build Service

- Continuous Integration
  - Practice of regularly merging code
  - Verified by automated build
- Jenkins
  - Open source
  - Used by the SESC build service

- The CCP-WSI code repository uses the SESC build service as a quality control measure.

- DEMO

Science & Technology Facilities Council

Required Software for remainder of workshop

- Software required:
  - g++
  - cppcheck
  - gdb (and optionally ddd)
  - gprof
  - valgrind
  - Kcachegrind

Please ask during the short morning break if you have trouble installing any of this software.

- To install, for example on Ubuntu:

```
sudo apt-get install g++ cppcheck gdb ddd binutils valgrind kcachegrind
```