# Contributing to PETSc is Easy!

Karl Rupp

rupp@iue.tuwien.ac.at

https://www.karlrupp.net/

Institute for Microelectronics, TU Wien

PETSc User Meeting 2018

Imperial College London, UK

June 4-6, 2018

**LIVE DEMO**
Let's enhance PETSc together!

*Debugging does not mean just trying a bunch of random things
with no understanding and hoping to figure out the problem.*

# In a Nutshell

### Create Pull Request in BitBucket

Fork the repository at `https://bitbucket.org/petsc/petsc/`

Create a new branch off master:
`git checkout -b myname/mybranchname`

Apply changes

Commit changes: `git commit -a`

Push changes: `git push -u origin myname/mybranchname`

Create pull request to master branch in main PETSc repository via webinterface
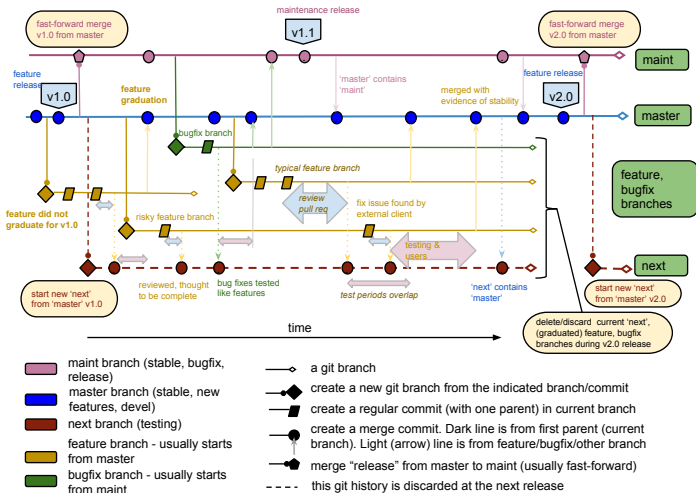
### Prefer GitHub?

You may also fork our mirror at GitHub

Repeat steps above

**What about all those branches in the repository?**
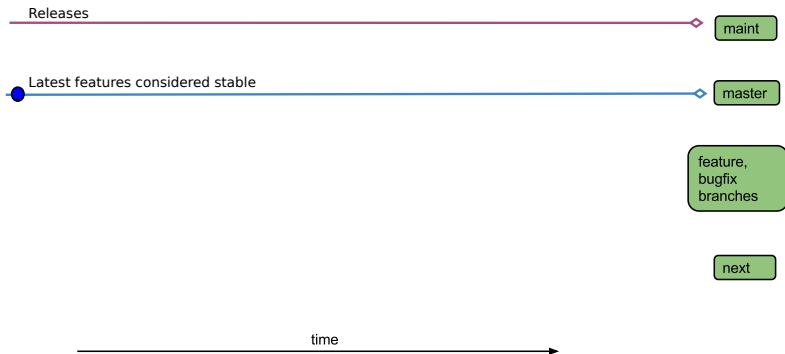
*git sucks!*

## PETSc's Workflow



*Git is the perl of revision control systems.*

# PETSc and Git



maint

master

feature,
bugfix
branches

next

time

*Oh, yes we all do need that push!*

Releases — maint

Latest features considered stable — master

feature, bugfix branches

next

time

*It's like we only offer the black belt for people who currently have no belts.*

feature release

v1.0

maint

master
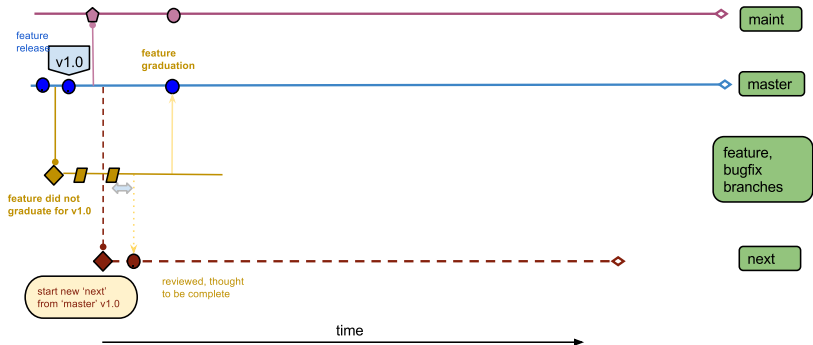
feature, bugfix branches

next

feature did not graduate for v1.0

time

*If it ain't in email or slashdot then I ain't read it :-)*
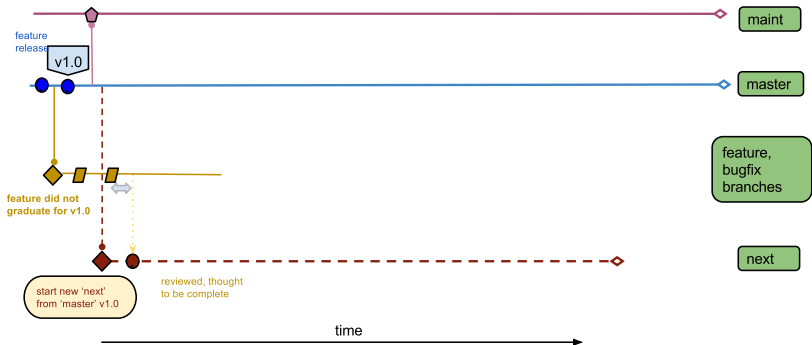
*I have now permanently blackholed all email dealing with cygwin.*

maint

feature release

v1.0

feature graduation

master

feature, bugfix branches

feature did not graduate for v1.0

next

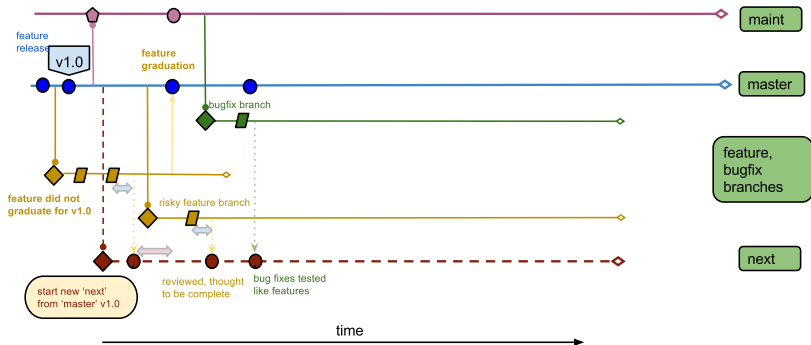start new 'next' from 'master' v1.0

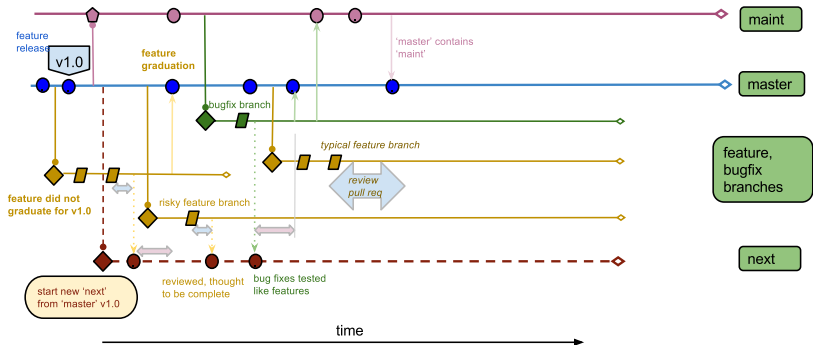reviewed, thought to be complete

time

*Peter, Matt gave some bad advice.*
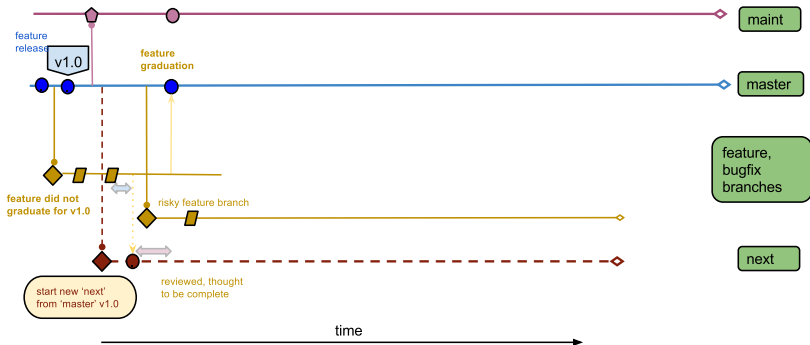
*Satish, please put this in next with the other six million branchs.*

# PETSc and Git

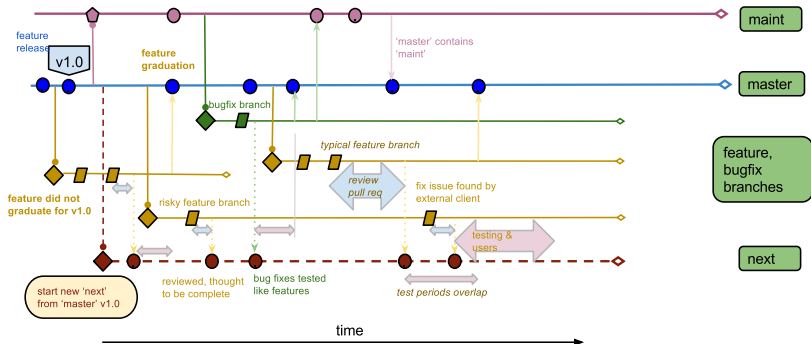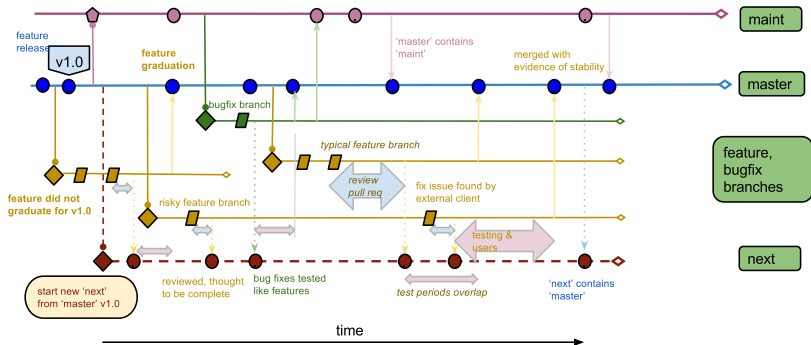*Don't tell anyone that I put a printf in here*

*I won't call the ICNTL() monstrosity in MUMPS an interface :-)*

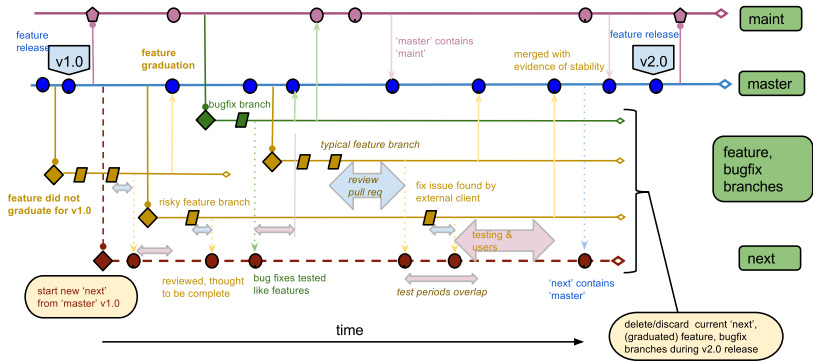*That is all I know about valgrind and it has saved my bacon many times.*

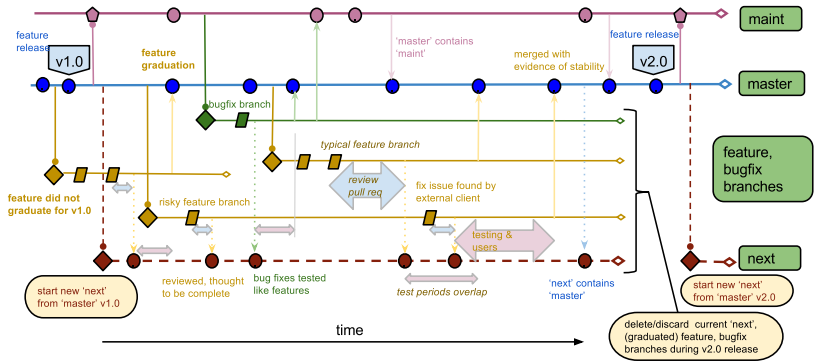*Stop right here. 99.9% of the time what you describe should not happen*

*Git is TeX, what we need is someone to come along and produce LaGit and thus make it usable.*

*Git commands are like irregular verbs in a foreign language:
there's no all-encompassing logic to them, you just have to memorize
which words you need for each thing you want to express*

## Conclusions

### PETSc can Help You

solve algebraic and DAE problems in your application area

rapidly develop efficient parallel code, can start from examples

develop new solution methods and data structures

debug and analyze performance

advice on software design, solution algorithms, and performance

```
petsc-{users,dev,maint}@mcs.anl.gov
```

### You can Help PETSc

report bugs and inconsistencies, or if you think there is a better way

tell us if the documentation is inconsistent or unclear

consider developing new algebraic methods as plugins, contribute if your idea works

*Thank you!*