Collaborative Tools 1
Project Management and Workflow

Mark Miesch

(JCSDA)

JEDI Academy – 4-7 June 2018

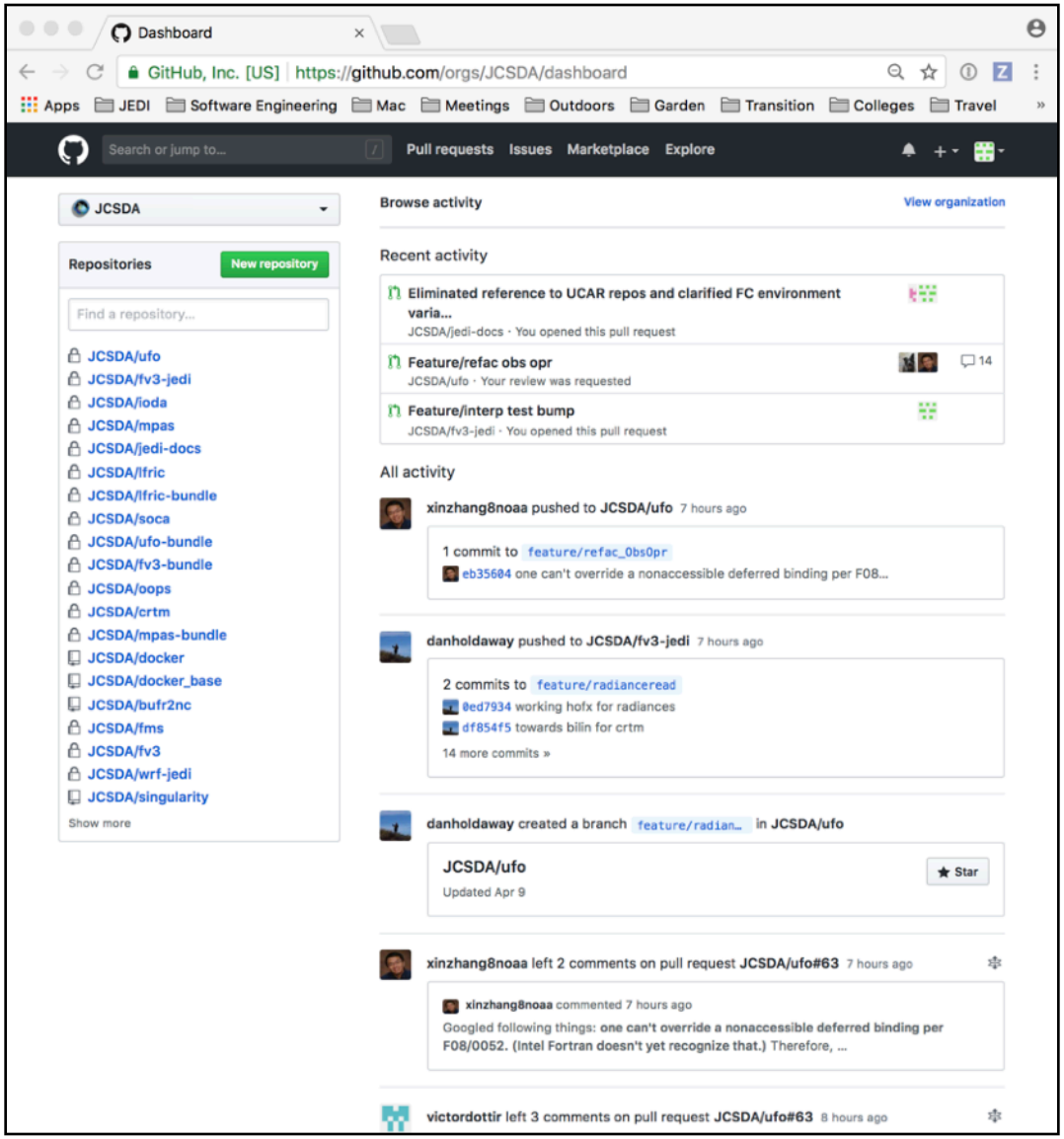
The Way of a JEDI

- ***Collaborative***
  - ***A Joint Center (JCSDA)***
    - ***Partners, collaborators, stakeholders, community***
  - ***A Joint Effort (JEDI)***
    - ***Distributed team of software developers, with varying objectives and time commitments***
- ***Agile***
  - ***Innovative***
  - ***Flexible and Limitless (future-proof)***
  - ***Responsive to users and developers***
  - ***Continuous delivery of functional software***

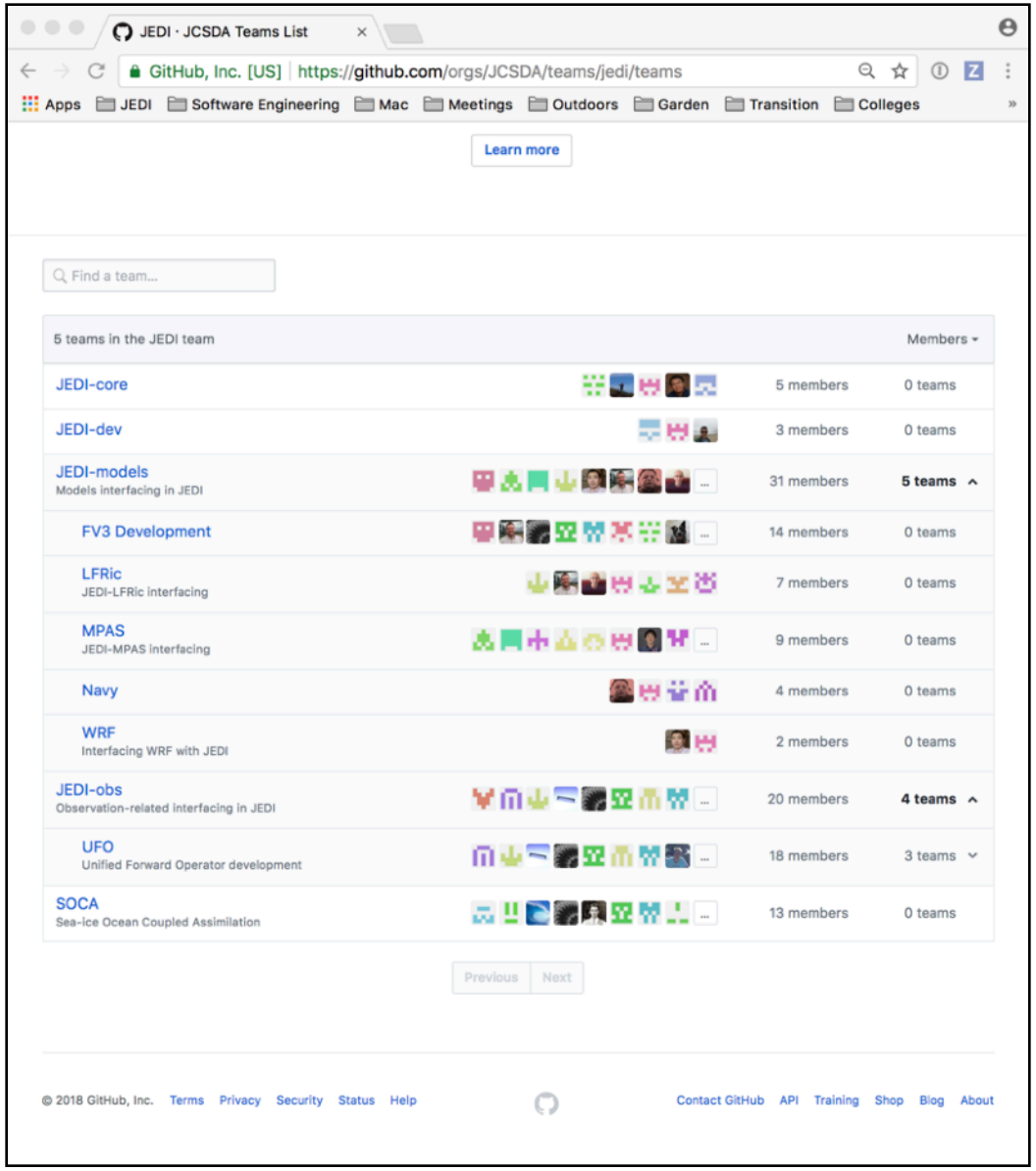
Outline

- ***git/GitHub***
  - ***Version control***
  - ***Enhancements and bug fixes immediately available to distributed community of developers***
  - ***Code review, issue tracking***
  - ***Community exports (Code distribution)***
    …and imports (ecbuild, eckit, fckit)
- ***Git-Flow***
  - ***Innovation***
  - ***Continuous Delivery***
- ***ZenHub***
  - ***Agile project management***
  - ***Enhances GitHub's issue tracking and code review functionality***

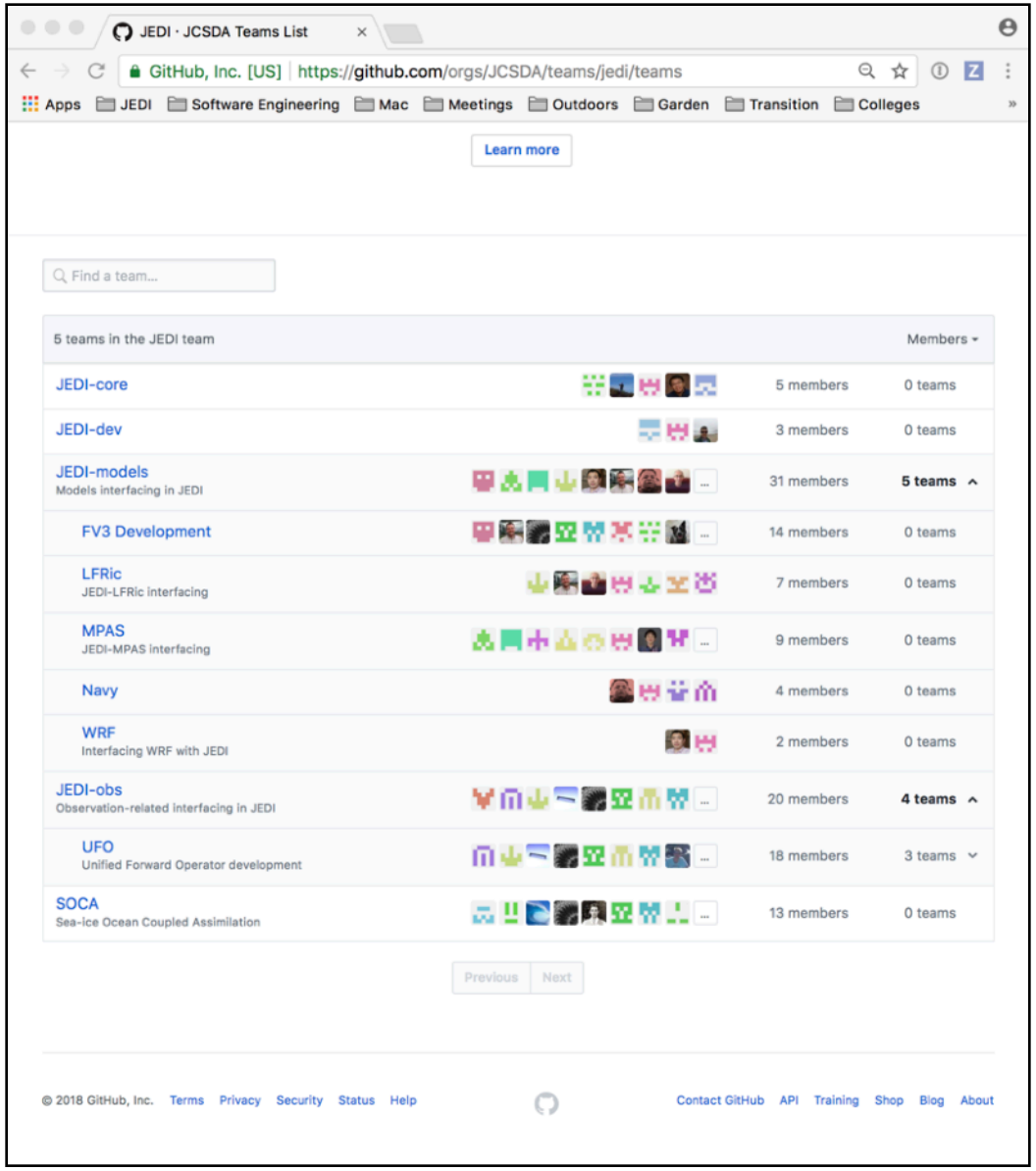
GitHub

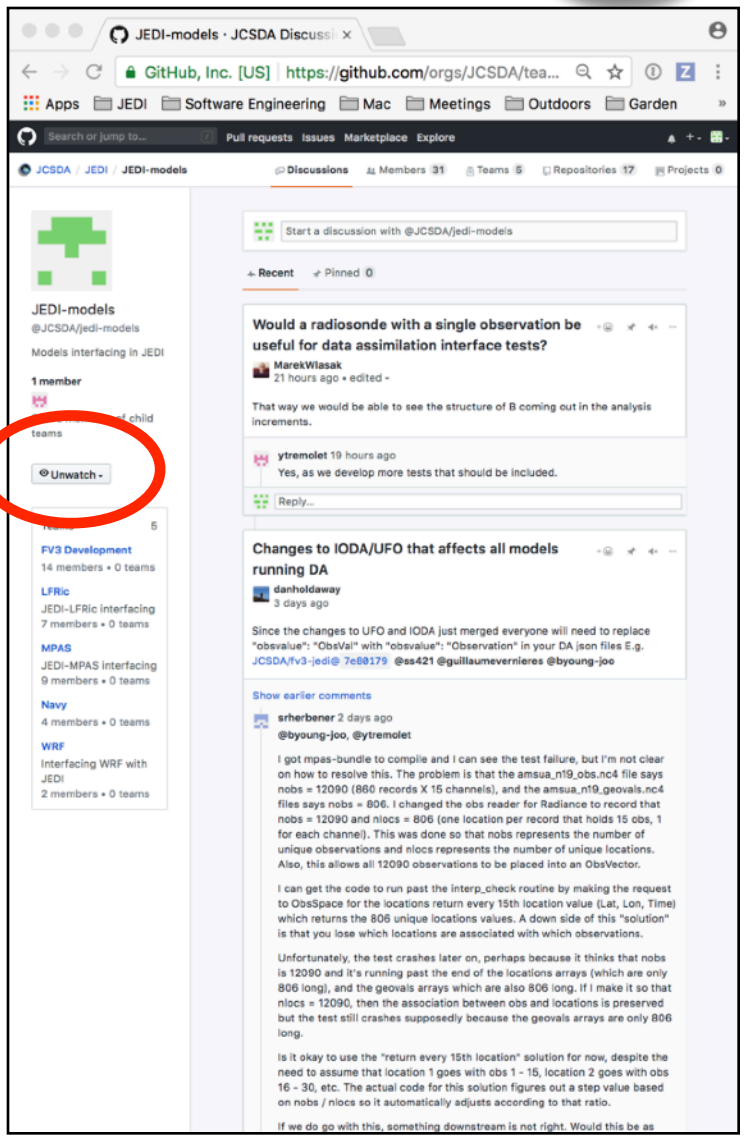

Dashboard
GitHub, Inc. [US] https://github.com/orgs/JCSDA/dashboard
Apps JEDI Software Engineering Mac Meetings Outdoors Garden Transition Colleges Travel
Search or jump to...
Pull requests Issues Marketplace Explore
JCSDA
Repositories
New repository
Find a repository...
JCSDA/ufo
JCSDA/fv3-jedi
JCSDA/ioda
JCSDA/mpas
JCSDA/jedi-docs
JCSDA/lfric
JCSDA/lfric-bundle
JCSDA/soca
JCSDA/ufo-bundle
JCSDA/fv3-bundle
JCSDA/oops
JCSDA/crtm
JCSDA/mpas-bundle
JCSDA/docker
JCSDA/docker_base
JCSDA/bufr2nc
JCSDA/fms
JCSDA/fv3
JCSDA/wrf-jedi
JCSDA/singularity
Show more
Browse activity
View organization
Recent activity
Eliminated reference to UCAR repos and clarified FC environment varia...
JCSDA/jedi-docs · You opened this pull request
Feature/refac obs opr
JCSDA/ufo · Your review was requested
14
Feature/interp test bump
JCSDA/fv3-jedi · You opened this pull request
All activity
xinzhang8noaa pushed to JCSDA/ufo 7 hours ago
1 commit to feature/refac_ObsOpr
eb35604 one can't override a nonaccessible deferred binding per F08...
danholdaway pushed to JCSDA/fv3-jedi 7 hours ago
2 commits to feature/radianceread
0ed7934 working hofx for radiances
df854f5 towards bilin for crtm
14 more commits »
danholdaway created a branch feature/radian... in JCSDA/ufo
JCSDA/ufo
Updated Apr 9
Star
xinzhang8noaa left 2 comments on pull request JCSDA/ufo#63 7 hours ago
xinzhang8noaa commented 7 hours ago
Googled following things: one can't override a nonaccessible deferred binding per F08/0052. (Intel Fortran doesn't yet recognize that.) Therefore, ...
victordottir left 3 comments on pull request JCSDA/ufo#63 8 hours ago

GitHub Teams



GitHub Teams





git/GitHub (JEDI tips)

- ***Work with JEDI bundles***
  - ✦***Clone bundle repo***
  - ✦***Let ecbuild do the rest***
  - ✦***If that doesn't work, read the README file***
  - ✦***Get in the habit of running make update after ecbuild***
  - ✦***Edit the CMakeLists.txt file to use your local version***

```
#ecbuild_bundle( PROJECT ufo    GIT "https://github.com/JCSDA/ufo.git"    BRANCH develop UPDATE )
ecbuild_bundle( PROJECT ufo SOURCE "~/jedi/src/ufo-bundle/ufo" )
```

- ***Cache your GitHub credentials***

```
git config --global credential.helper 'cache --timeout=3600'
```

Git-LFS

- *LFS = Large File service*
  - *Increases GitHub size limits for individual files from 100 MB to 2GB*
  - *Cumulative storage purchased in 50 GB data packs*
  - *Used for anything that isn't code (data files, restart files, etc)*
- *Transparent to the user*
  - *When you push to GitHub, any files that are tracked by LFS will go to a remote server (the LFS Store)*
  - *The GitHub repo will only contain a pointer to that file*
  - *When you fetch/pull/clone an LFS-enabled repo from GitHub, LFS will check to see if you have the large files on your computer (local LFS cache). If not, it will retrieve them from the LFS Store as needed.*

Using Git-LFS

1) Extension to git

- brew install git-lfs

2) See if git-lfs is already enabled for that repo

- git lfs track

3) If not already sufficient, then add appropriate tracking patterns

- git lfs install # only if step 2 returns nothing
- git lfs track *.nc4

4) Add your large files to the repo

5) Make sure your files and patterns are tracked by git

- git add .gittattributes
- git add * # new files

6) commit, push, pull, fetch, clone and proceed as you would with any other repo

Git-Flow

Git Flow is:

- ***A Philosophy***
  - ***Optimal for Agile Software Development***
    - ***Innovation***
    - ***Continuous Delivery***
- ***A Working Principle***
  - ***Enforcement of branch naming conventions soon to come***
- ***An Application (extension to git)***
  - ***Already installed in Singularity Container***
  - **brew install git-flow-avh # (Mac)**
  - **sudo apt-get install git-flow # (linux)**
  - **https://github.com/petervanderdoes/gitflow-avh**

The Git-Flow Manifesto

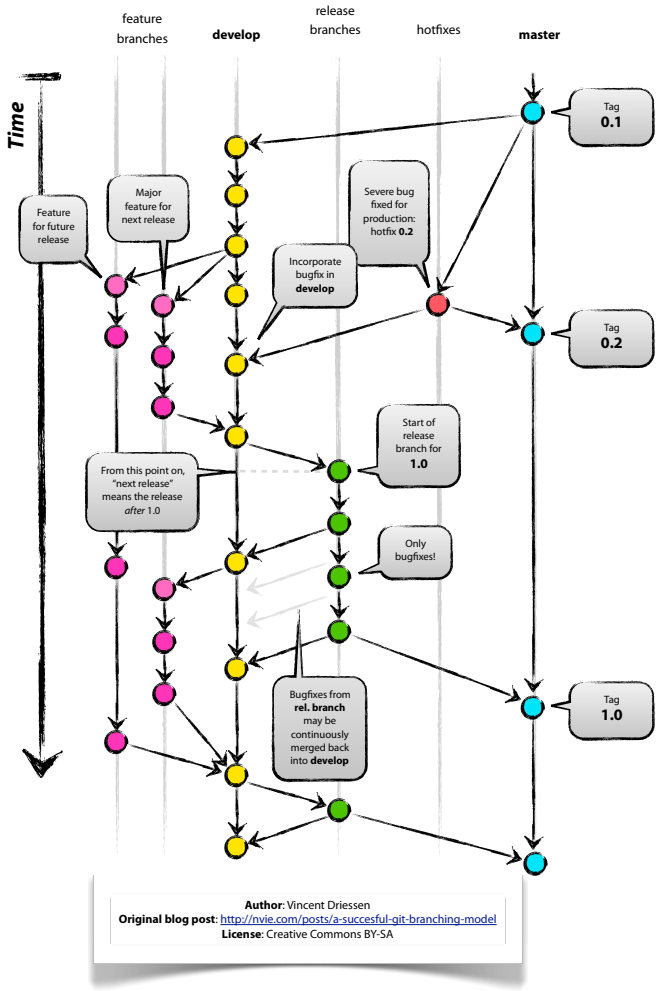
http://nvie.com/posts/a-successful-git-branching-model/

Vincent Driessen (2010)

Google

git branching model

Highly Recommended!



The Git-Flow Manifesto: Takaways

- ***master is for releases only***
- ***develop***
  - ***Not ready for pubic consumption but compiles and passes all tests***
- ***Feature branches***
  - ***Where most development happens***
  - ***Branch off of develop***
  - ***Merge into develop***
- ***Release branches***
  - ***Branch off of develop***
  - ***Merge into master and develop***
- ***Hotfix***
  - ***Branch off of master***
  - ***Merge into master and develop***
- ***Bugfix***
  - ***Branch off of develop***
  - ***Merge into develop***

Life Cycle of a Feature branch

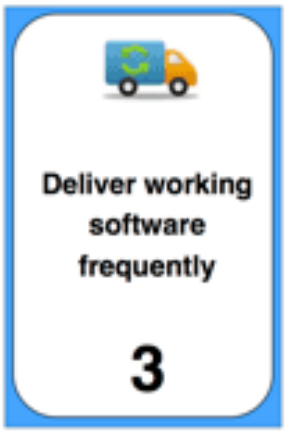
1) Enable git flow for the repo
   - git flow init -d
2) Start the feature branch
   - git flow feature start newstuff
   - Creates a new branch called feature/newstuff that branches off of develop
3) Push it to GitHub for the first time
   - Make changes and commit them locally
   - git flow feature publish newstuff
4) Additional (normal) commits and pushes as needed
   - git commit -a
   - git push
5) Bring it up to date with develop (to minimize big changes on the ensuing pull request)
   - git checkout develop
   - git pull origin develop
   - git checkout feature/newstuff
   - git merge develop
6) Finish the feature branch (don't use git flow feature finish)
   - Do a pull request on GitHub from feature/newstuff to develop
   - When successfully merged the remote branch will be deleted
   - git remote update -p
   - git branch -d feature/newstuff

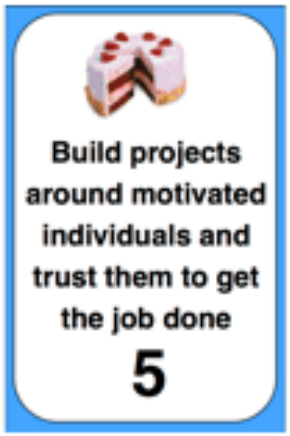
Git-Flow: Lingering Questions

- ***What about forks?***
  - ✦ ***For now, developers can work off the central repo***
  - ✦ ***As the project grows, each parter/collaborator institution will maintain a fork (merge with central repo as needed)***
  - ✦ ***Forking may also be useful for public releases***
- ***What if I can't install git-flow?***
  - ✦ ***Just be sure to use the proper naming and branching conventions***
    - feature/mybranch
    - release/mybranch
    - bugfix/mybranch
    - hotfix/mybranch
- ***More?***

Agile Software Development

- ***12 Agile Principles***





https://nomad8.com/

Agile Software Development

- ***12 Agile Principles***

Git-Flow helps with many of these
For the rest, we have ZenHub

Early and continuous delivery of valuable software

1 ✔

Welcome changing requirements even late in development

2 ✔

Deliver working software frequently

3 ✔

Business people and developers working together daily

4

Build projects around motivated individuals and trust them to get the job done

5 ✔

The most effective method of conveying information is face-to-face conversation

6

Working software is the primary measure of progress

7 ✔

Sustainable development: maintain a constant pace indefinitely

8

Continuous attention to technical excellence

9 ✔

Simplicity: maximize the amount of work not done

10 ✔

Teams self-organize

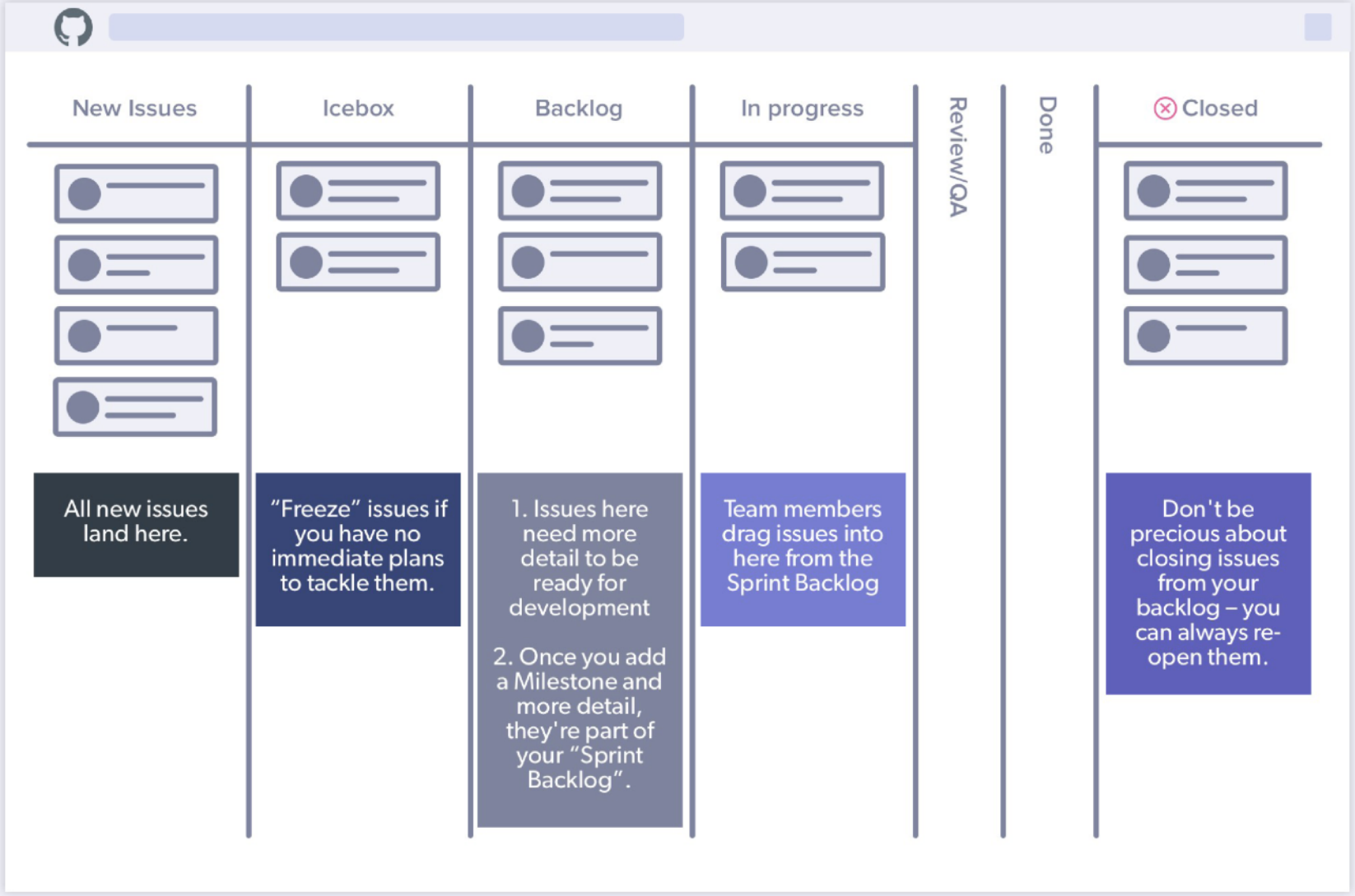
11

Teams regularly reflect and adjust behaviour

12 ✔

https://nomad8.com/

ZenHub (Sample board)



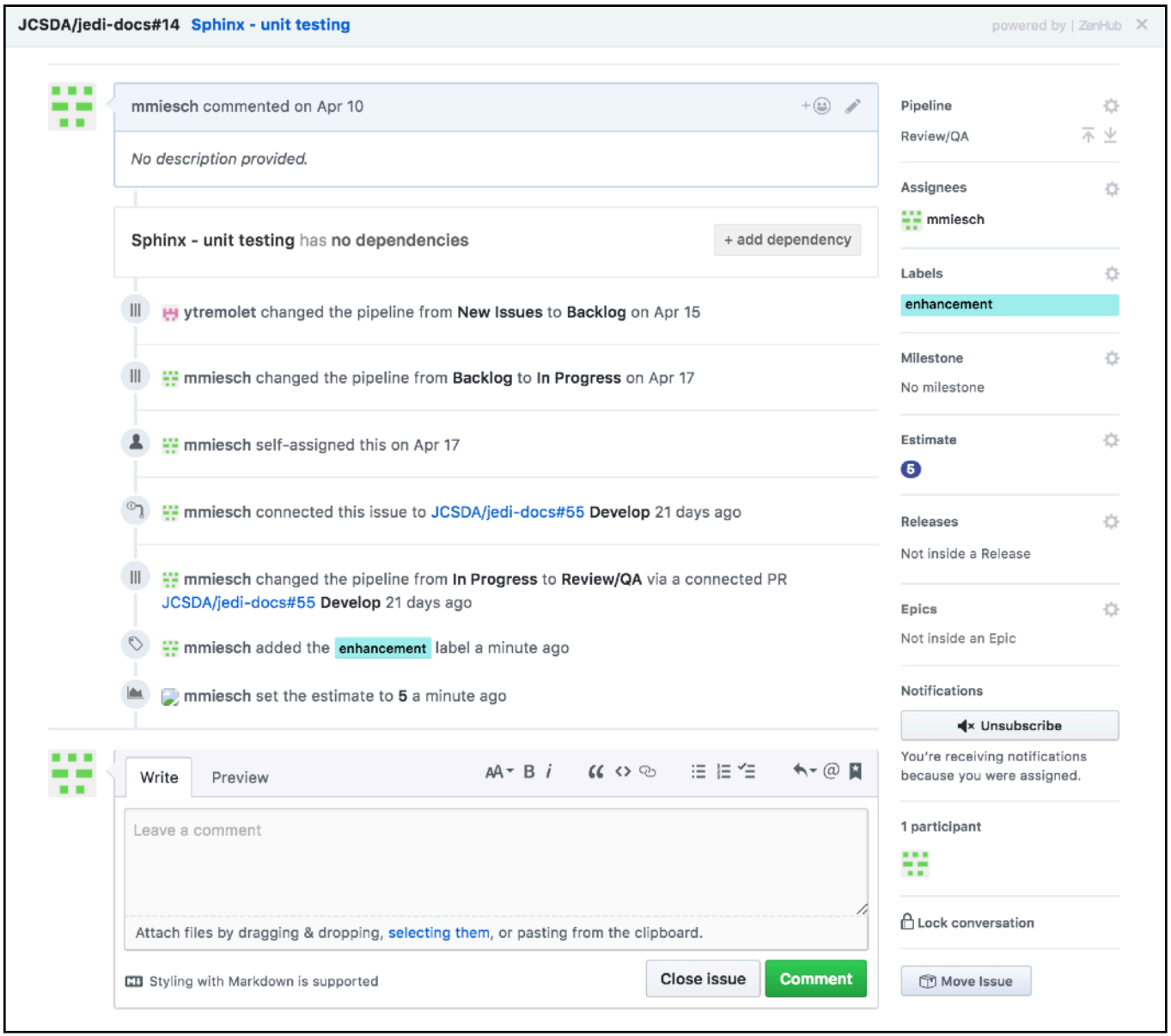
ZenHub Features

- ***Customizable Project boards***
  - ✦ ***Prioritize and organize tasks***
  - ✦ ***Reviews/Feedback***
  - ✦ ***Sprints (Milestones) and Epics***
- ***Closely integrated with GitHub***
  - ✦ ***Access boards directly from GitHub repos***
  - ✦ ***ZenHub tasks are GitHub issues and vice versa***
- ***Tasks/Issues***
  - ✦ ***Assign up to 10 individuals***
  - ✦ ***Labels, difficulty estimates, etc.***
  - ✦ ***Can be linked to pull requests***
  - ✦ ***Markdown supported (boldface, checklists…)***
- ***Monitoring progress***
  - ✦ ***Burndown charts***
  - ✦ ***Velocity tracking***
  - ✦ ***Release reports***
    - ***Time estimate to deliver a specified set of features***

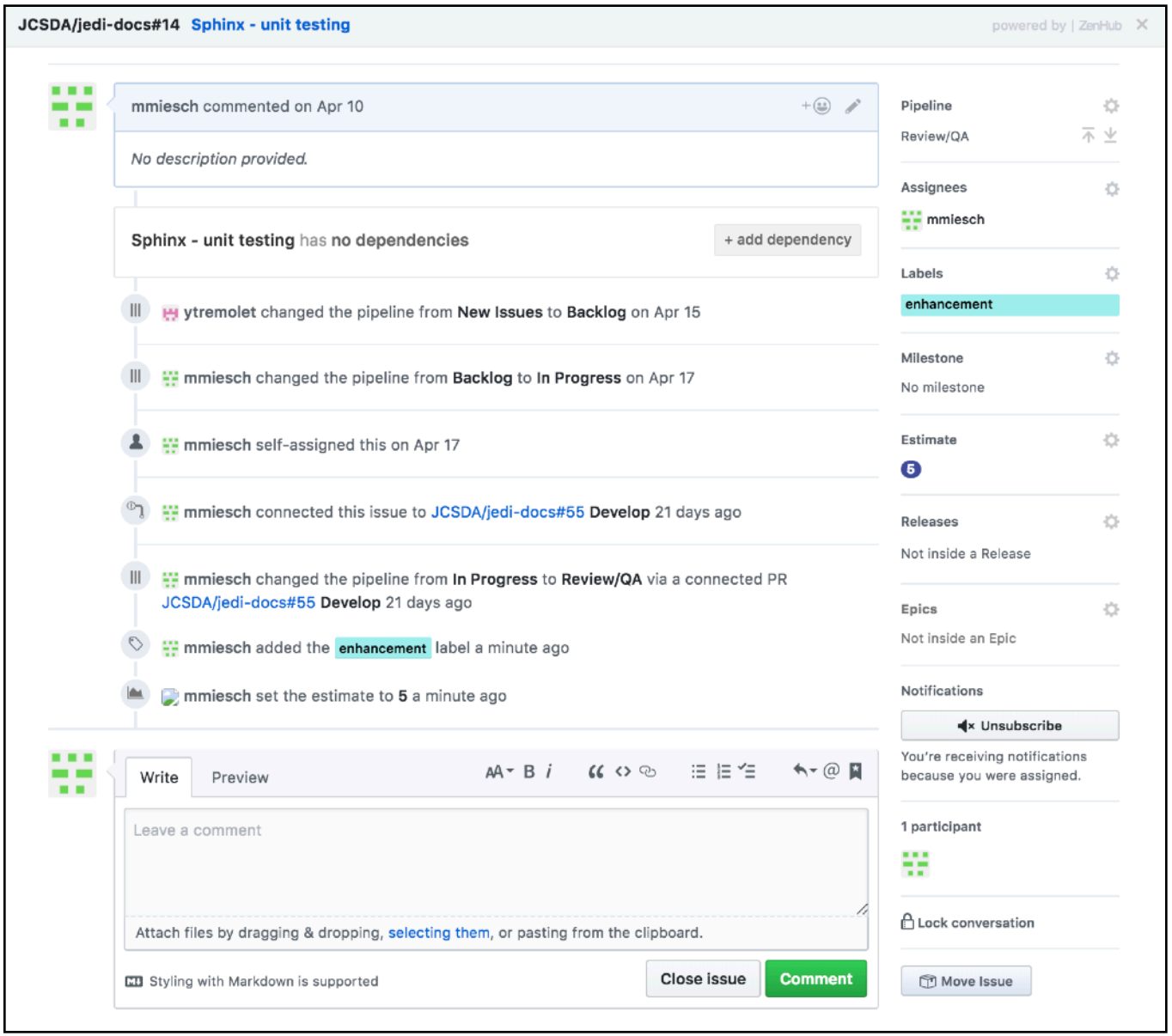
ZenHub Pipelines

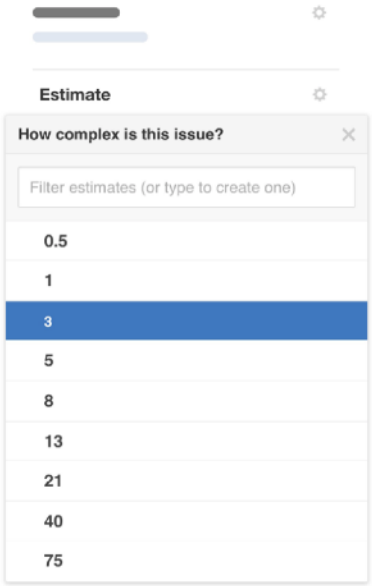
- ***New Issues***
  - ***Default landing spot***
  - ***Issues should not stay here long***
- ***Backlog***
  - ***Main "To Do" List***
  - ***Arrange in order of priority (reviewed regularly by teams)***
- ***IceBox***
  - ***Low-priority items that should be done at some point but do not require immediate attention***
- ***In Progress***
  - ***Lets others know what you are doing to promote collaboration and avoid redundancy***
- ***Review/QA***
  - ***Solicit feedback before you mark something as…***
- ***Closed***

ZenHub Issues / Tasks



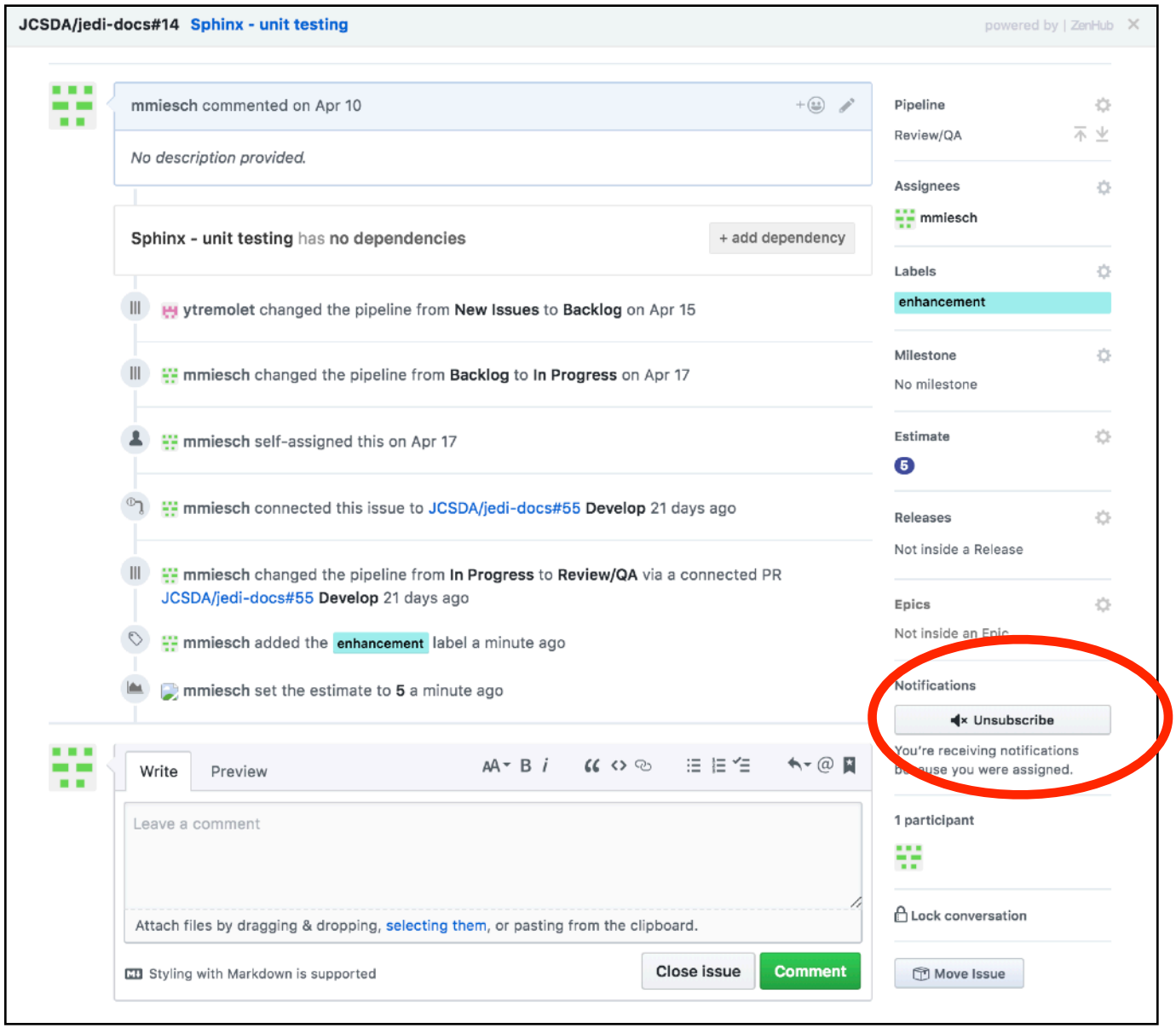
ZenHub Issues/Tasks

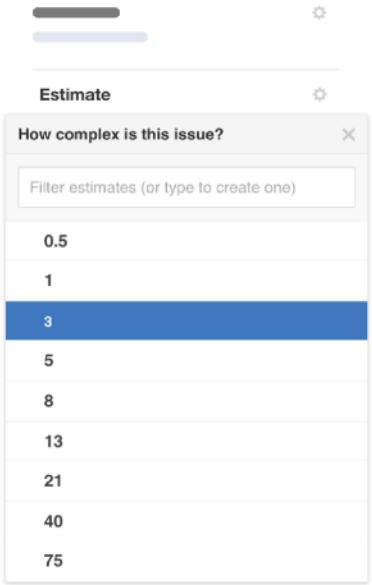




Suggestion:
1 unit = 1/2 day
dedicated work

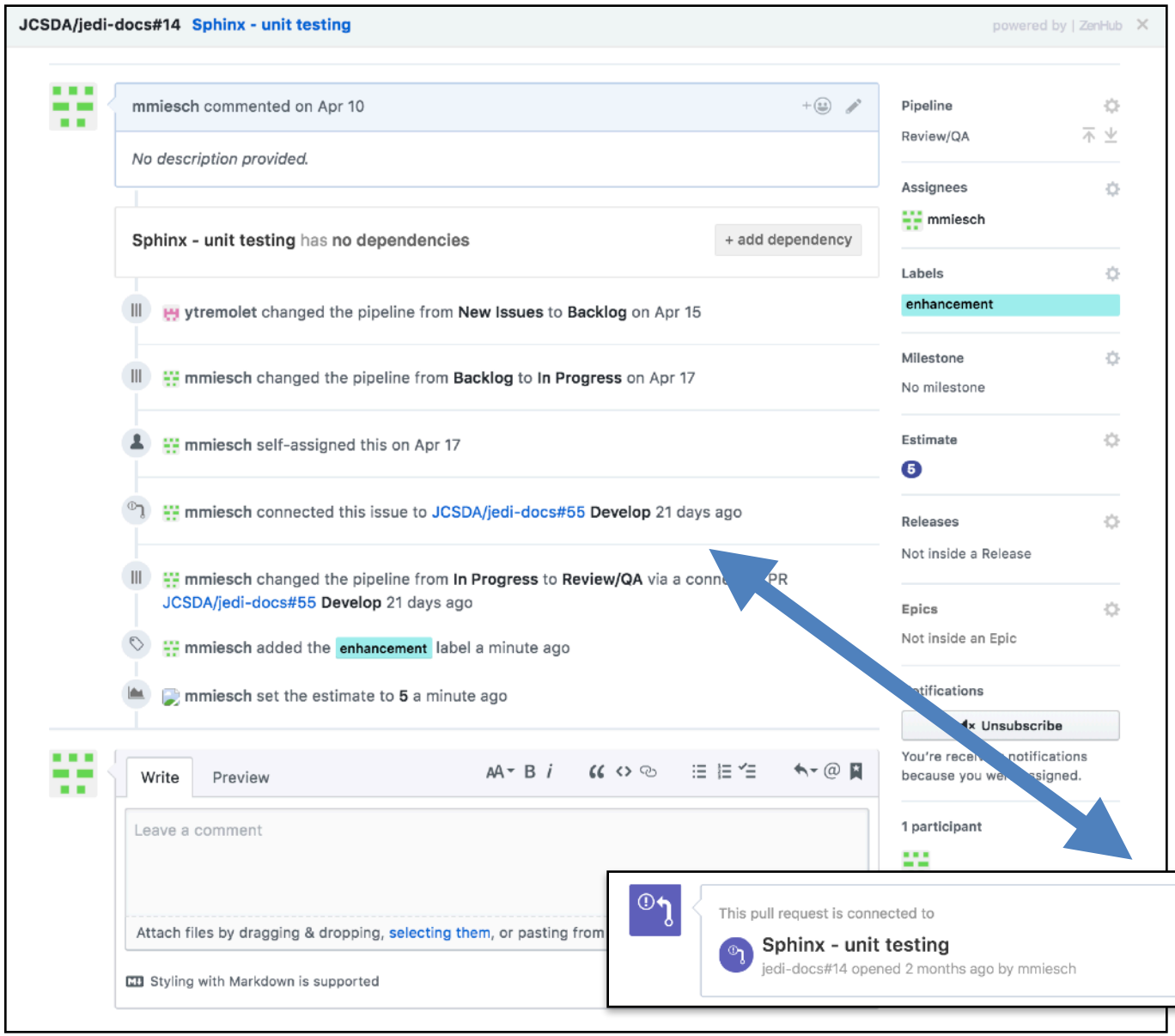
ZenHub Issues/Tasks

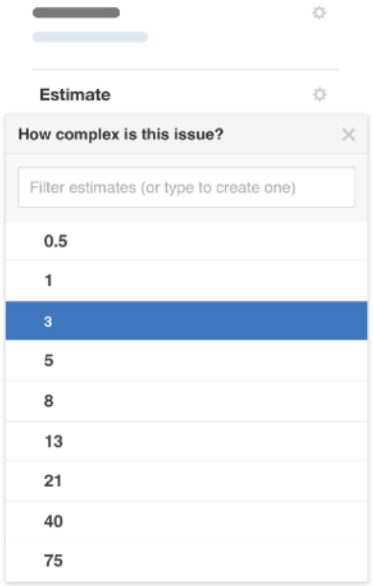




Suggestion:
1 unit = 1/2 day
dedicated work

ZenHub Issues/Tasks





Suggestion:
1 unit = 1/2 day
dedicated work

ZenHub: Milestones and Epics

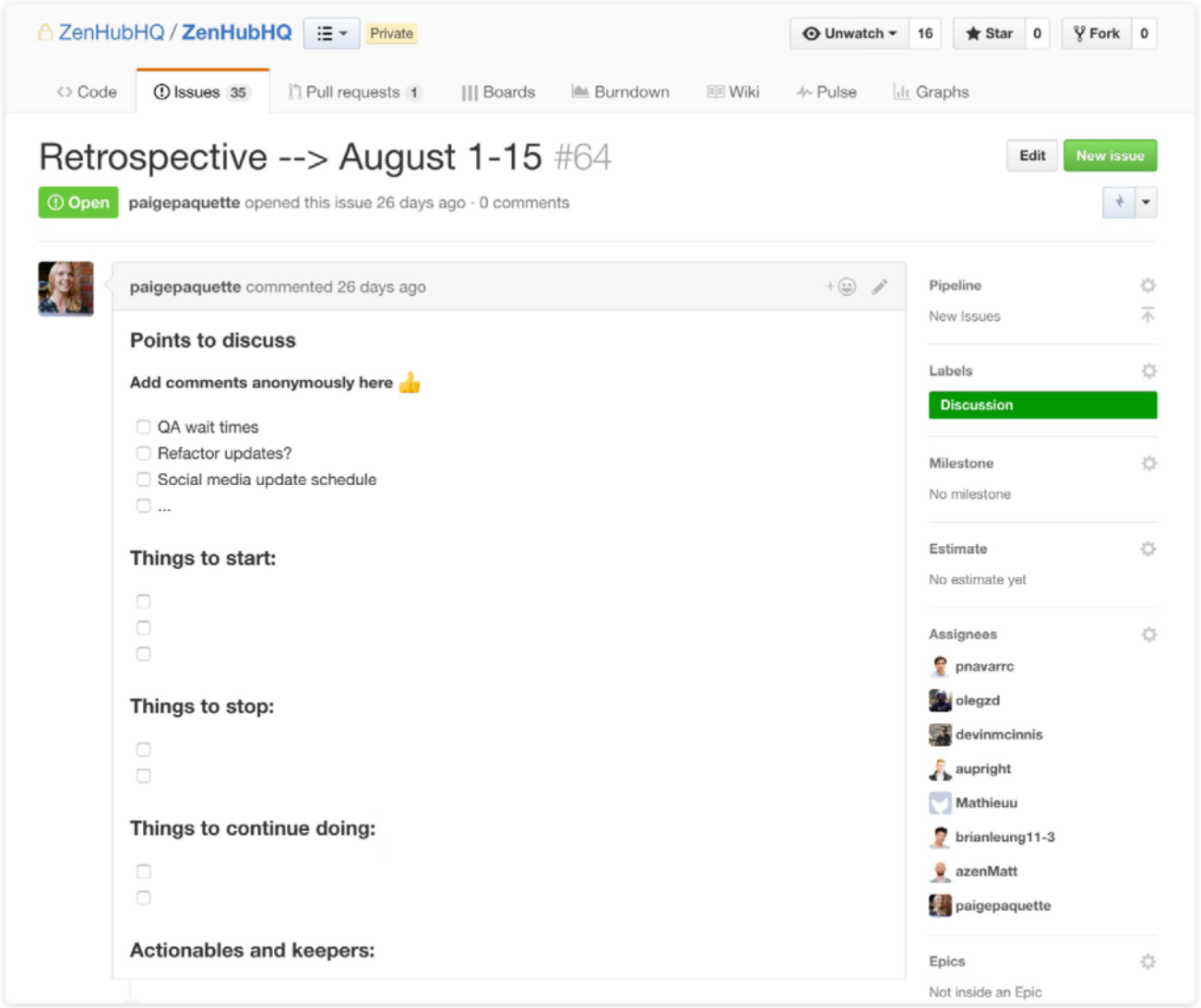
- ***Milestones (Sprints)***
  - ***Short-term (~ 2 weeks)***
  - ***Focused work, often on 1-2 repos***
  - ***Deliverables = specific functionality/features***

- ***Epics***
  - ***Long-term (indefinite)***
  - ***Typically span multiple repos***
  - ***Deliverables = releases, guiding vision***

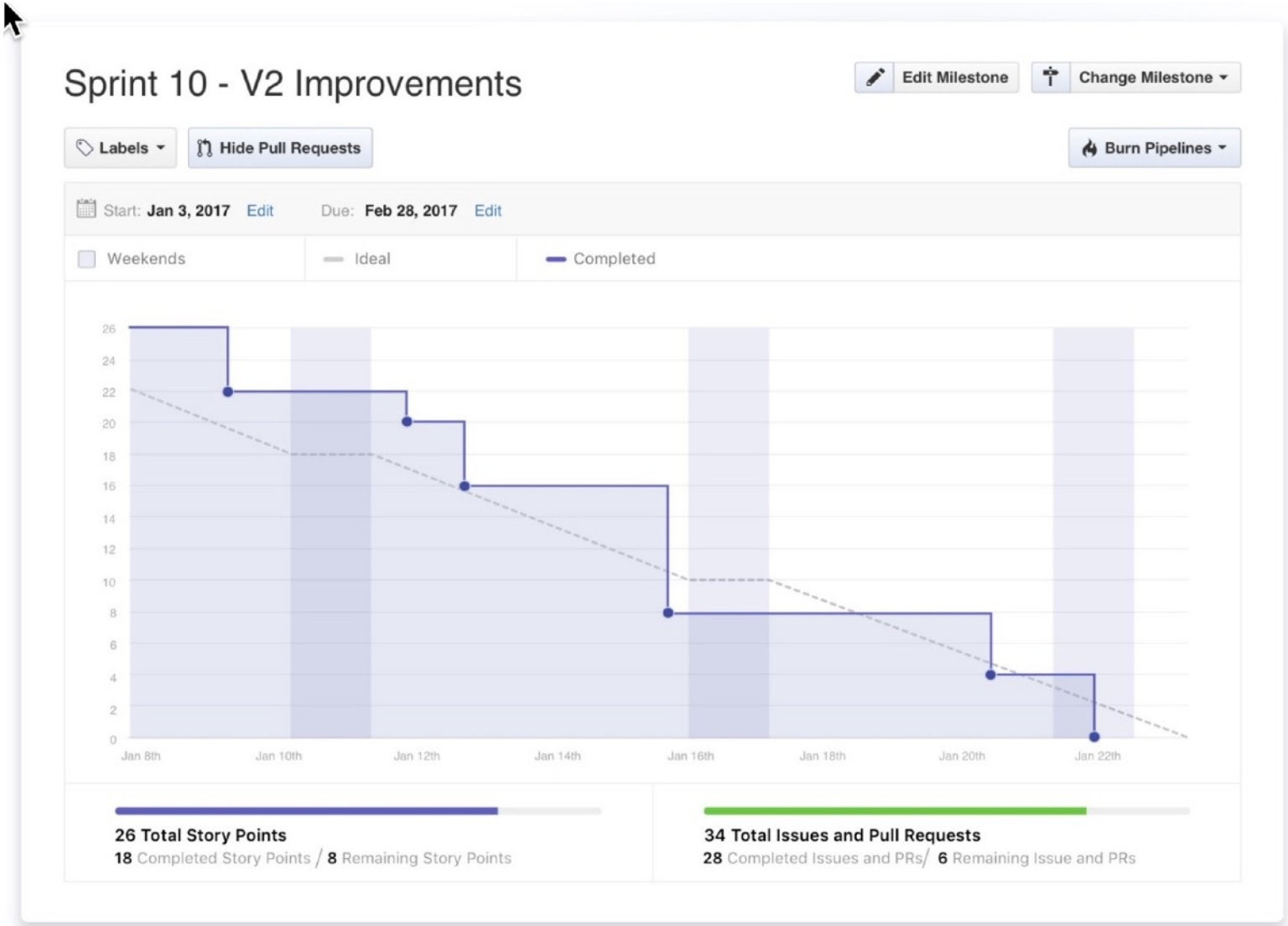
Project boards include filters to view only issues associated with Milestones, Epics or other attributes (assignee, label, repo, release…)

ZenHub: Sprint Retrospective

Sprint Retrospectives and other agile workflow components (Sprint Review, Release Planning, etc) are best done face-to-face, but one could in principle dedicate an issue or a pipeline to solicit further perspectives



ZenHub: Burndown chart



ZenHub: Release Report

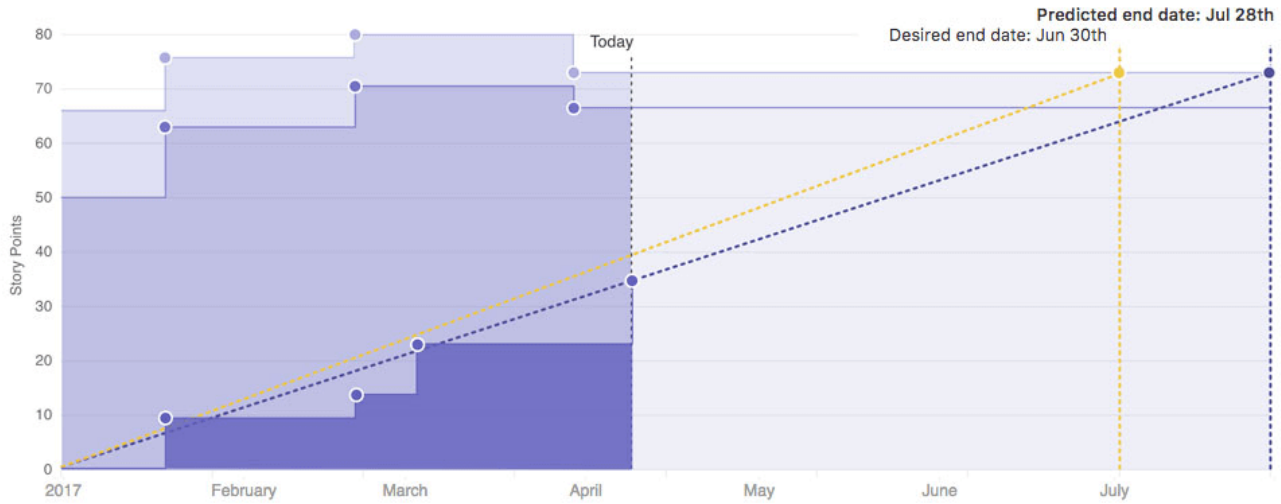
Project X

Combined projects across Q1 and Q2, working towards the next chapter of AppX.

Start date: **Jan 1st, 2017** Desired end date: **Jun 30th, 2017**

Labels

Release report



The predicted end date is **Jul 28th**

The predicted end date will be 28 days behind the desired end date of Jun 30th

Total issues	**7**	**Total story points**	**30**
Completed	3	Completed	21
Remaining	4	Remaining	9

Resources: ZenHub/GitHub

ZenHub Guides
https://www.zenhub.com/guides

Extensive GitHub documentation & tutorials
https://help.github.com

Lots of Great Github Cheat Sheets
https://education.github.com/git-cheat-sheet-education.pdf
https://jan-krueger.net/git-cheat-sheet-extended-edition
https://patrickzahnd.ch/uploads/git-transport-v1.png

Resources: Git-Flow

JEDI Git Flow page
https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/developer/developer_tools/getting-started-with-gitflow.html

The Git Flow manifesto (all you need to know about the philosophy):
http://nvie.com/posts/a-successful-git-branching-model/

Git Flow cheat sheet:
https://danielkummer.github.io/git-flow-cheatsheet/

Git avh (a fork of the original, with added features):
https://github.com/petervanderdoes/gitflow-avh

Atlassian git-flow tutorial (philosophy and application):
https://www.atlassian.com/git/tutorials/comparing-workflows/gitflow-workflow

Resources: Git-LFS

JEDI Git-LFS page
https://jointcenterforsatellitedataassimilation-jedi-docs.readthedocs-hosted.com/en/latest/developer/developer_tools/gitlfs.html

GitHub's Help page:
https://help.github.com/articles/about-git-large-file-storage/

Tutorial:
https://github.com/git-lfs/git-lfs/wiki/Tutorial

Installation? Already installed in the JEDI singularity container

Binaries available for download on:
https://git-lfs.github.com

Or, on a Mac:

brew install git-lfs