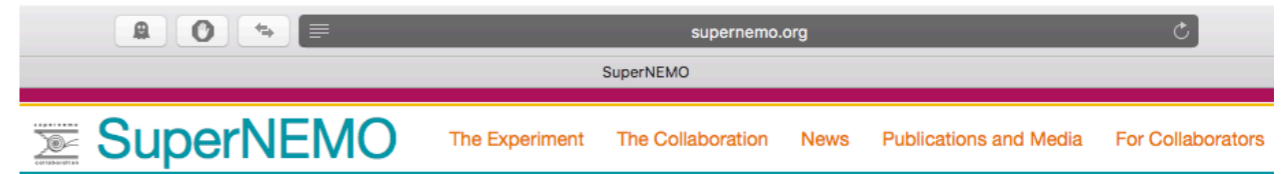


Docker in the SuperNEMO Experiment

Ben Morgan

SuperNEMO

- Search for Neutrinoless Double Beta Decay
- O(100-200) researchers
- Offline software is based on in-house framework from LPC-Caen (also used in GANIL and industrial partnerships)
- Core dependencies: C++11 compiler, Boost, ROOT6, Geant4, Qt5
- **1-2FTE on Software/Computing, split over ~4 people (0.5 from me)**



How about a SuperNEMO presentation or poster at your conference?

[Request a speaker](#)

Searching for Neutrinoless Double Beta Decay

The NEMO (Neutrino Ettore Majorana Observatory) collaboration is an international physics effort including the experiment SuperNEMO and its predecessor, NEMO-3.

The [SuperNEMO demonstrator module](#) is currently being assembled at the [LSM](#) underground lab, located in the Fréjus tunnel near Modane, France. NEMO-3, also at the LSM, ran from 2003-11. Its rich repository of data is still being analysed today.

Both SuperNEMO and NEMO-3 are designed to study extremely rare double-beta decay processes, and in particular are looking for evidence of [neutrinoless double beta decay](#). This is a rare type of radioactive decay which has been predicted, but has never been observed. If this process was seen, it would prove that neutrinos were their own antiparticles, which could be a clue to the matter-antimatter asymmetry in the universe.

Theme based on [jekyll/minima](#)

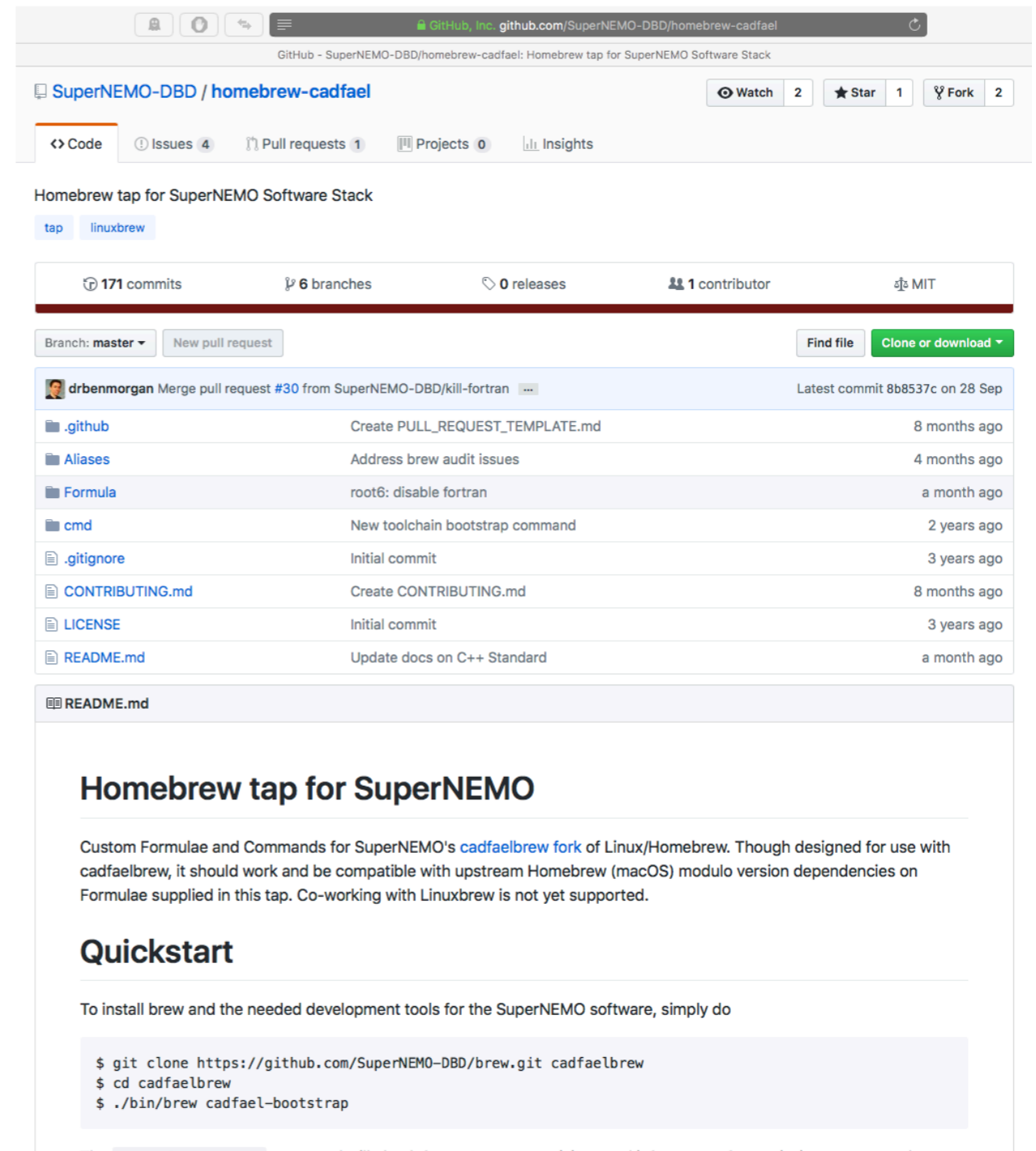
Copyright © 2017, All Rights Reserved, SuperNEMO Collaboration

What is a “Small” Experiment Nowadays?

- LHC/Intensity Frontier experiments at $O(1k-4k)$ people
- Wide range of HEP/Neutrino/Dark Matter/etc experiments below this scale, $O(100-500)$ people: small (-ish?)
- In software/packaging terms, means fewer FTEs for development/deployment, re-use of existing software/resources very important (but not always done!)

Package Management in SuperNEMO

- Migrated to Linux/Homebrew in 2015
- Choice based on simplicity and availability
- Follow upstream packages except for core dependencies like Geant4, ROOT, where a “Tap” is supplied
- Binary “bottles” not used yet - FTE limits, plus understanding use of custom glibc.
- But found we did need a binary distribution mechanism...



GitHub - SuperNEMO-DBD/homebrew-cadfael: Homebrew tap for SuperNEMO Software Stack

SuperNEMO-DBD / homebrew-cadfael

Watch 2 Star 1 Fork 2

Code Issues 4 Pull requests 1 Projects 0 Insights

Homebrew tap for SuperNEMO Software Stack

tap linuxbrew

171 commits 6 branches 0 releases 1 contributor MIT

Branch: master New pull request Find file Clone or download

drbenmorgan Merge pull request #30 from SuperNEMO-DBD/kill-fortran Latest commit 8b8537c on 28 Sep

.github	Create PULL_REQUEST_TEMPLATE.md	8 months ago
Aliases	Address brew audit issues	4 months ago
Formula	root6: disable fortran	a month ago
cmd	New toolchain bootstrap command	2 years ago
.gitignore	Initial commit	3 years ago
CONTRIBUTING.md	Create CONTRIBUTING.md	8 months ago
LICENSE	Initial commit	3 years ago
README.md	Update docs on C++ Standard	a month ago

README.md

Homebrew tap for SuperNEMO

Custom Formulae and Commands for SuperNEMO's [cadfaelbrew](#) fork of Linux/Homebrew. Though designed for use with [cadfaelbrew](#), it should work and be compatible with upstream Homebrew (macOS) modulo version dependencies on Formulae supplied in this tap. Co-working with Linuxbrew is not yet supported.

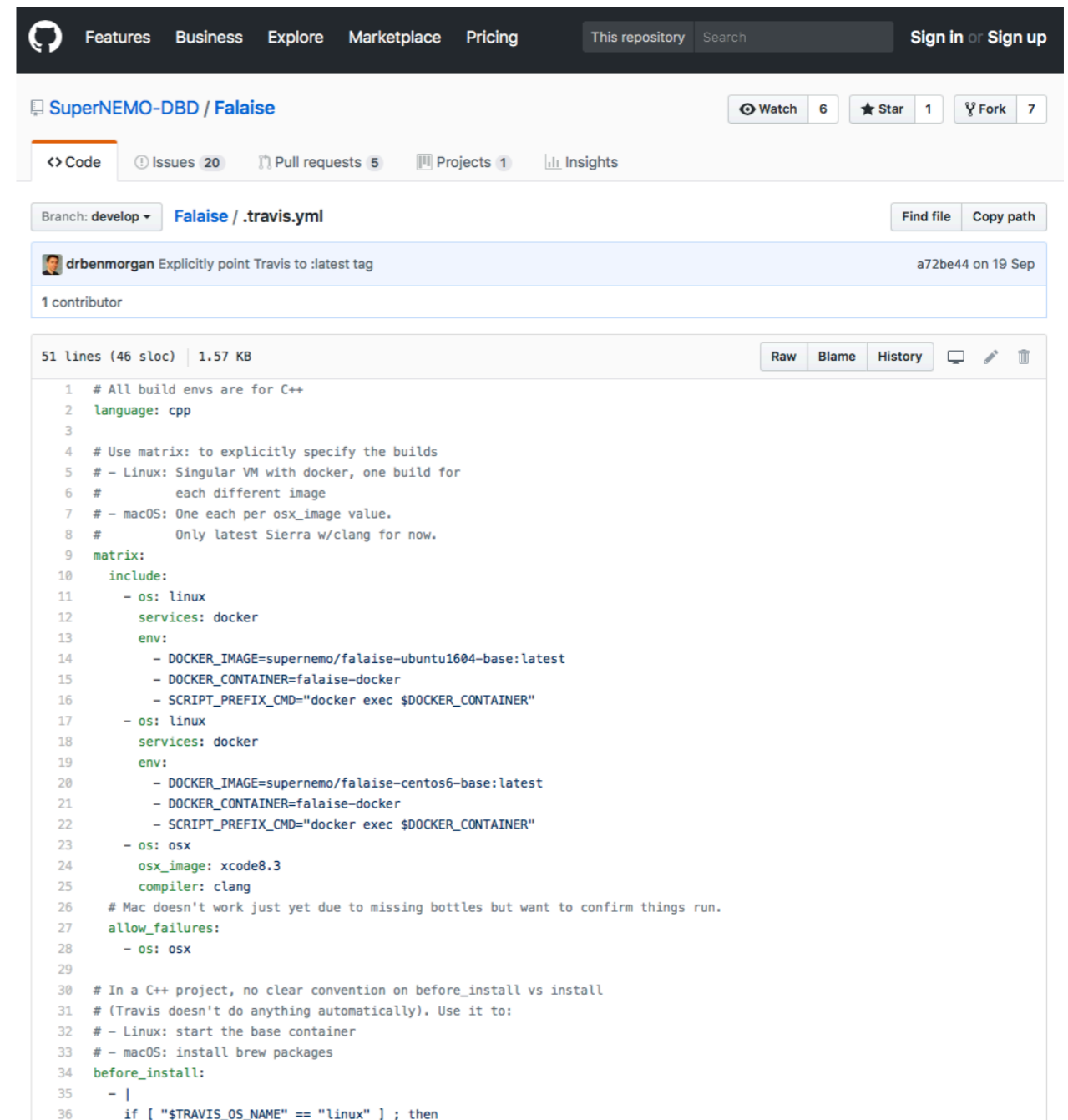
Quickstart

To install brew and the needed development tools for the SuperNEMO software, simply do

```
$ git clone https://github.com/SuperNEMO-DBD/brew.git cadfaelbrew
$ cd cadfaelbrew
$ ./bin/brew cadfael-bootstrap
```

Need for Docker in SuperNEMO

- Migrated development from internal SVN to Git/Hub earlier this year
- Use of Pull Request model => easier testing, potential for “easy” CI with Travis
- **But... dependencies build from source, not reasonable on Travis.**
- Solution for Linux - create docker image(s) for base system(s) + Brew'd dependencies

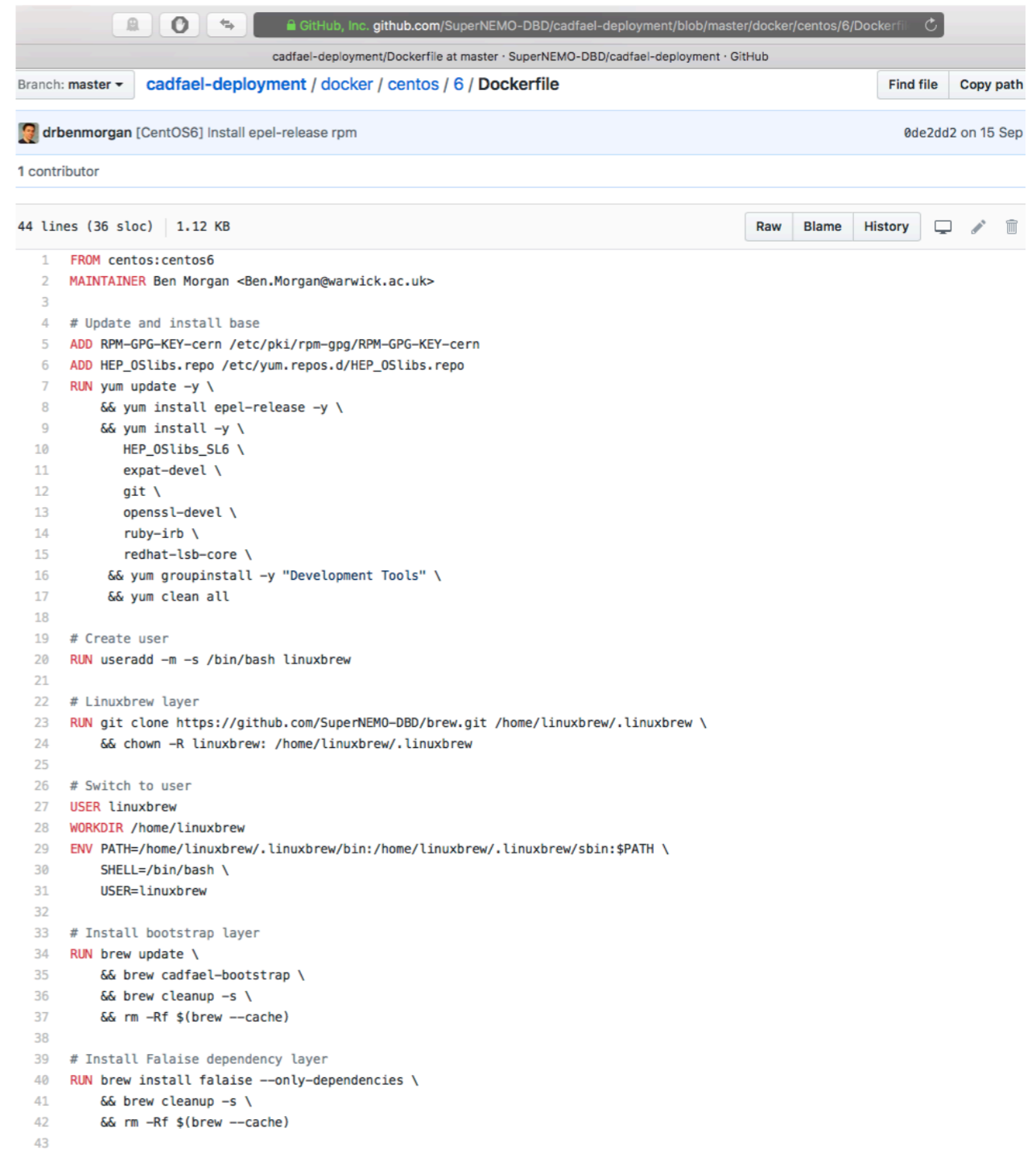


The screenshot shows the GitHub interface for the repository 'SuperNEMO-DBD / Falaise'. The file '.travis.yml' is selected, showing its content. The file is 51 lines long, with 46 sloc and 1.57 KB. It defines a Travis CI build matrix for C++ projects, supporting Linux and macOS. The matrix includes configurations for Ubuntu, CentOS, and macOS, each with specific Docker images and environment variables. The file also includes a 'before_install' section for Linux that starts the base container.

```
1 # All build envs are for C++
2 language: cpp
3
4 # Use matrix: to explicitly specify the builds
5 # - Linux: Singular VM with docker, one build for
6 #   each different image
7 # - macOS: One each per osx_image value.
8 #   Only latest Sierra w/clang for now.
9 matrix:
10 include:
11   - os: linux
12     services: docker
13     env:
14       - DOCKER_IMAGE=supernemo/falaise-ubuntu1604-base:latest
15       - DOCKER_CONTAINER=falaise-docker
16       - SCRIPT_PREFIX_CMD="docker exec $DOCKER_CONTAINER"
17   - os: linux
18     services: docker
19     env:
20       - DOCKER_IMAGE=supernemo/falaise-centos6-base:latest
21       - DOCKER_CONTAINER=falaise-docker
22       - SCRIPT_PREFIX_CMD="docker exec $DOCKER_CONTAINER"
23   - os: osx
24     osx_image: xcode8.3
25     compiler: clang
26 # Mac doesn't work just yet due to missing bottles but want to confirm things run.
27 allow_failures:
28   - os: osx
29
30 # In a C++ project, no clear convention on before_install vs install
31 # (Travis doesn't do anything automatically). Use it to:
32 # - Linux: start the base container
33 # - macOS: install brew packages
34 before_install:
35   - |
36     if [ "$TRAVIS_OS_NAME" == "linux" ]; then
```

Building Containers

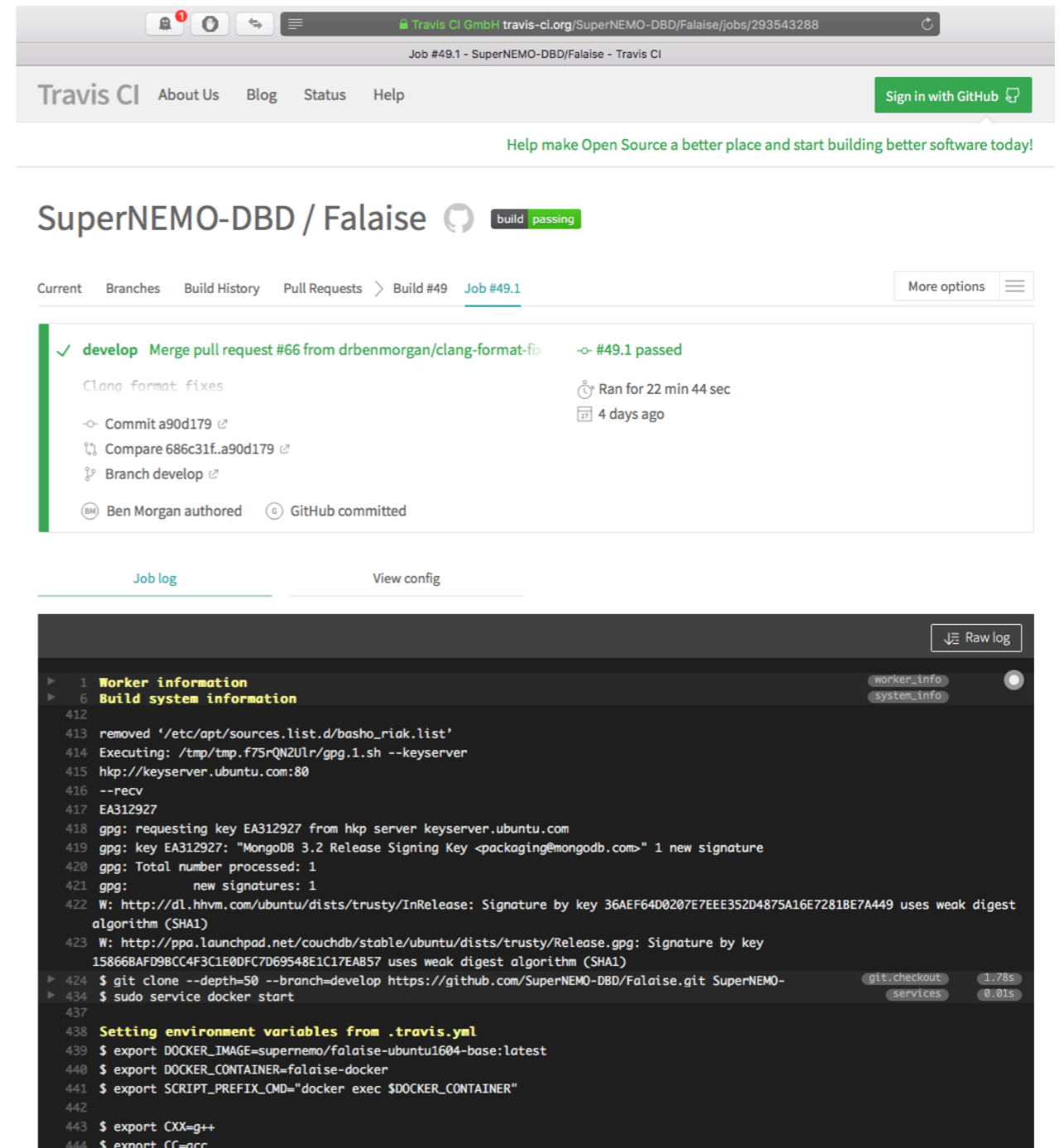
- Use CentOS6/7 and Ubuntu 14/16LTS base images
- Layers added:
 - System rpm/deb packages
 - Brew'd Compiler/Dev tools
 - Brew'd Build/Test packages
- Named/Tagged by underlying distro and “latest” for use in CI
- Hosted on Docker Hub, but don't yet use their CI to build images



```
1 FROM centos:centos6
2 MAINTAINER Ben Morgan <Ben.Morgan@warwick.ac.uk>
3
4 # Update and install base
5 ADD RPM-GPG-KEY-cern /etc/pki/rpm-gpg/RPM-GPG-KEY-cern
6 ADD HEP_OSlibs.repo /etc/yum.repos.d/HEP_OSlibs.repo
7 RUN yum update -y \
8     && yum install epel-release -y \
9     && yum install -y \
10     HEP_OSlibs_SL6 \
11     expat-devel \
12     git \
13     openssl-devel \
14     ruby-irb \
15     redhat-lsb-core \
16     && yum groupinstall -y "Development Tools" \
17     && yum clean all
18
19 # Create user
20 RUN useradd -m -s /bin/bash linuxbrew
21
22 # Linuxbrew layer
23 RUN git clone https://github.com/SuperNEMO-DBD/brew.git /home/linuxbrew/.linuxbrew \
24     && chown -R linuxbrew: /home/linuxbrew/.linuxbrew
25
26 # Switch to user
27 USER linuxbrew
28 WORKDIR /home/linuxbrew
29 ENV PATH=/home/linuxbrew/.linuxbrew/bin:/home/linuxbrew/.linuxbrew/sbin:$PATH \
30     SHELL=/bin/bash \
31     USER=linuxbrew
32
33 # Install bootstrap layer
34 RUN brew update \
35     && brew cadfael-bootstrap \
36     && brew cleanup -s \
37     && rm -Rf $(brew --cache)
38
39 # Install Falaise dependency layer
40 RUN brew install falaise --only-dependencies \
41     && brew cleanup -s \
42     && rm -Rf $(brew --cache)
43
```

Containers in Use

- Images are large, 1-4GB!
- Travis doesn't cache images, but in practice, pulling the images ~10-15% of each build/test job.
- Only using Ubuntu 16/Centos 6 images to save resource.
- macOS builds still need brew'd bottles!
- If we bottled on Linux as well, Docker might not be needed.

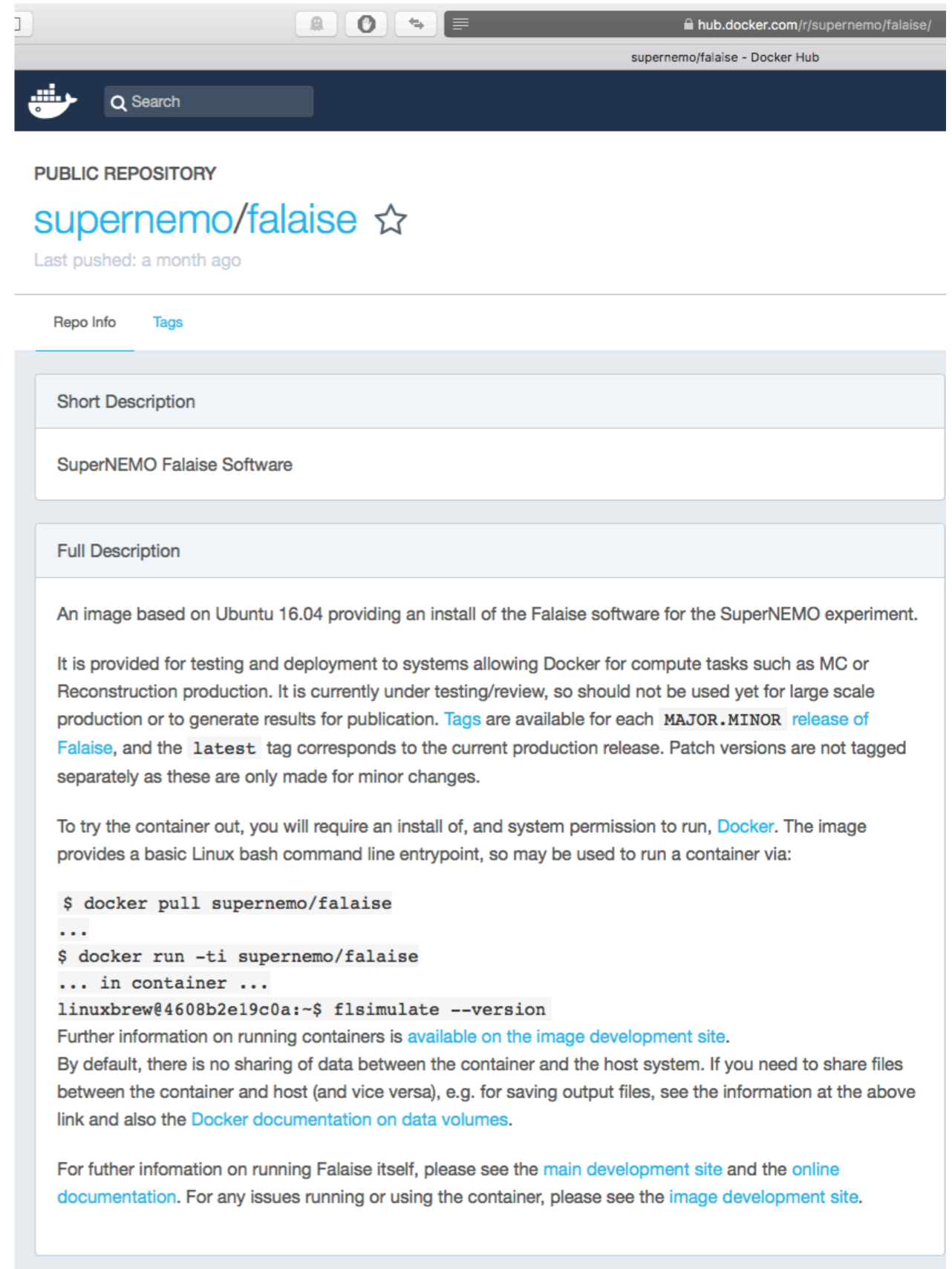


The screenshot shows a Travis CI build job for the repository SuperNEMO-DBD / Falaise. The job is passing and is identified as Job #49.1. The build log shows the following steps:

```
1 Worker information
6 Build system information
412 removed '/etc/apt/sources.list.d/basho_riak.list'
414 Executing: /tmp/tmp.f75rQN2U1r/gpg.1.sh --keyserver
415 hkp://keyserver.ubuntu.com:80
416 --recv
417 EA312927
418 gpg: requesting key EA312927 from hkp server keyserver.ubuntu.com
419 gpg: key EA312927: "MongoDB 3.2 Release Signing Key <packaging@mongodb.com>" 1 new signature
420 gpg: Total number processed: 1
421 gpg:      new signatures: 1
422 W: http://dl.hhvm.com/ubuntu/dists/trusty/InRelease: Signature by key 36AEF64D0207E7EEE352D4875A16E7281BE7A449 uses weak digest algorithm (SHA1)
423 W: http://ppa.launchpad.net/couchdb/stable/ubuntu/dists/trusty/Release.gpg: Signature by key 158668AFD9BCC4F3C1E0DFC7D69548E1C17EAB57 uses weak digest algorithm (SHA1)
424 $ git clone --depth=50 --branch=develop https://github.com/SuperNEMO-DBD/Falaise.git SuperNEMO-
434 $ sudo service docker start
437
438 Setting environment variables from .travis.yml
439 $ export DOCKER_IMAGE=supernemo/falaise-ubuntu1604-base:latest
440 $ export DOCKER_CONTAINER=falaise-docker
441 $ export SCRIPT_PREFIX_CMD="docker exec $DOCKER_CONTAINER"
442
443 $ export CXX=g++
444 $ export CC=gcc
```

Containers for Deployment

- Direct use of “brew install” works for most users
- Still, get images of current release of software for free
 - Just an additional layer
 - Based on Ubuntu 16LTS for smallest size/most modern toolchain
- Not used in production as we do not have access to Docker enable compute resources - there if we need it.



The screenshot shows the Docker Hub interface for the repository `supernemo/falaise`. The page is titled "PUBLIC REPOSITORY" and includes a search bar and navigation tabs for "Repo Info" and "Tags". The "Short Description" section states "SuperNEMO Falaise Software". The "Full Description" section provides detailed information about the image, including its base OS (Ubuntu 16.04), its purpose for testing and deployment, and instructions on how to run the container. It includes a code block with terminal commands and links to further documentation.

hub.docker.com/r/supernemo/falaise/
supernemo/falaise - Docker Hub

PUBLIC REPOSITORY
supernemo/falaise ☆
Last pushed: a month ago

Repo Info Tags

Short Description

SuperNEMO Falaise Software

Full Description

An image based on Ubuntu 16.04 providing an install of the Falaise software for the SuperNEMO experiment.

It is provided for testing and deployment to systems allowing Docker for compute tasks such as MC or Reconstruction production. It is currently under testing/review, so should not be used yet for large scale production or to generate results for publication. [Tags](#) are available for each `MAJOR.MINOR` [release of Falaise](#), and the `latest` tag corresponds to the current production release. Patch versions are not tagged separately as these are only made for minor changes.

To try the container out, you will require an install of, and system permission to run, [Docker](#). The image provides a basic Linux bash command line entrypoint, so may be used to run a container via:

```
$ docker pull supernemo/falaise
...
$ docker run -ti supernemo/falaise
... in container ...
linuxbrew@4608b2e19c0a:~$ flsimulate --version
```

Further information on running containers is [available on the image development site](#).

By default, there is no sharing of data between the container and the host system. If you need to share files between the container and host (and vice versa), e.g. for saving output files, see the information at the above link and also the [Docker documentation on data volumes](#).

For further information on running Falaise itself, please see the [main development site](#) and the [online documentation](#). For any issues running or using the container, please see the [image development site](#).

Open Questions (from this Docker newbie)

- Even with docker, binary packaging is still useful, so important that Package Manager(s) support this.
- How do/could/should Docker and CVMFS work together?
- Docker on multiuser systems (e.g. university desktops) seemingly a security no-no. Singularity to the rescue?
- Best practices for layering? Single release per image? Spack views? HSF/Framework “Starter Kit” Image?