

OpenFOAM コントリビュート活動

稲葉竜一 [1†] 松原 大輔 [2] @tkoyama010[1]
[1]OpenFOAM-jp [2] オープン CAE 勉強会

OpenFOAM Contributing Activities

Ryuichi INABA[*†] Daisuke MATSUBARA[**] @tkoyama010[*]
[*]OpenFOAM-jp [**]OpenCAE Local user group

Abstract

Keywords: GitHub, git, Pull Request, OpenFOAM, develop

1. はじめに

現状日本からの OpenFOAM へのコントリビュートは少ない．そこで有志により”OpenFOAM-jp” というグループを立ち上げ，本家の開発やドキュメントの翻訳，チュートリアルの作成などのコントリビュート活動を始めた．本報告ではその活動内容などについて発表する．

2. Foundation 版と ESI 版の違い

OpenFOAM は，Imperial Collage で開発された商用 CFD コード FOAM(Field Operation And Manipulation) が，2004 年に OpenFOAM と名前を変えてオープンソースとしてリリースされたことから始まる．[1][2] 2011 年には OpenFOAM を運営している OpenCFD 社が SGI 社に買収され，またその 2 社により OpenFOAM の商標を譲渡した OpenFOAM Foundation Inc. を設立した．これ以降 OpenFOAM は OpenFOAM Foundation Inc. より開発と配布が行われており，2019 年 11 月現在 OpenFOAM-V7 を最新版とする Fork が”Foundation 版”[3][4] と呼ばれている．

一方で OpenFOAM には多くの Fork が存在する．2012 年以降は ESI 社が OpenCFD 社の買収という形から開発に加わるようになり，2016 年には OpenFOAM-v3.0 から Fork した OpenFOAM-v3.0+ をリリースした．2019 年 11 月時点ではこの最新版は OpenFOAM-v1906 であり，この Fork が”ESI 版”[5][6] もしくは”plus 版”と呼ばれる．

本報告ではこの Foudation 版と ESI 版の 2 つについてコントリビュートの流れなどを報告する．

3. ESI 版コントリビュート方法

ESI 版の開発は GitLab で行われている．[6] このページからレポジトリを clone することができ，これをそのままコンパイルすることで次期リリース予定の ESI 版 OpenFOAM を使用することができる．またここには開発途中の branch が多くあり，将来的に追加されるであろう機能をチェックすることもできる．

図 1に示す GitLab のページの右上のボタンからこの OpenFOAM-plus のレポジトリに登録することもでき，これにより issue の作成や書き込みをすることができるようになる．例えばバグを見つけた場合や OpenFOAM への要望がある場合には，issue を作成することで開発者へ周知することができ，必要と判断された場合には開発者によりコードの修正や作成が行われる．図 2には実際に作成した issue の例を示す．内容としてはソルバーの Discription の中のスペルミスを指摘した簡単なものであるが，数日以内に@mark 氏により該当箇所の修正が行われていることが確認できる．このような小さなコントリビュートの積み重ねが OSS である OpenFOAM を精錬していくこととなる．

一方で，push や Pull Request などのコードの変更を行うための権限は制限されている．このため，ESI 版 OpenFOAM のコードにコントリビュートする手段としては，開発メンバーに加わらない限りは issue の作成も

† *E-mail address of corresponding author: inabower@gmail.co.jp*

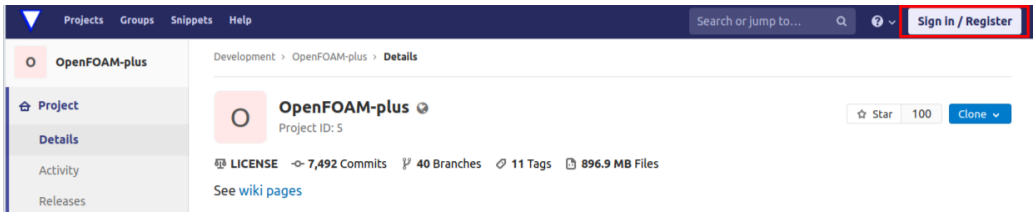


Fig. 1: A register button of OpenFOAM-plus's GitLab page.

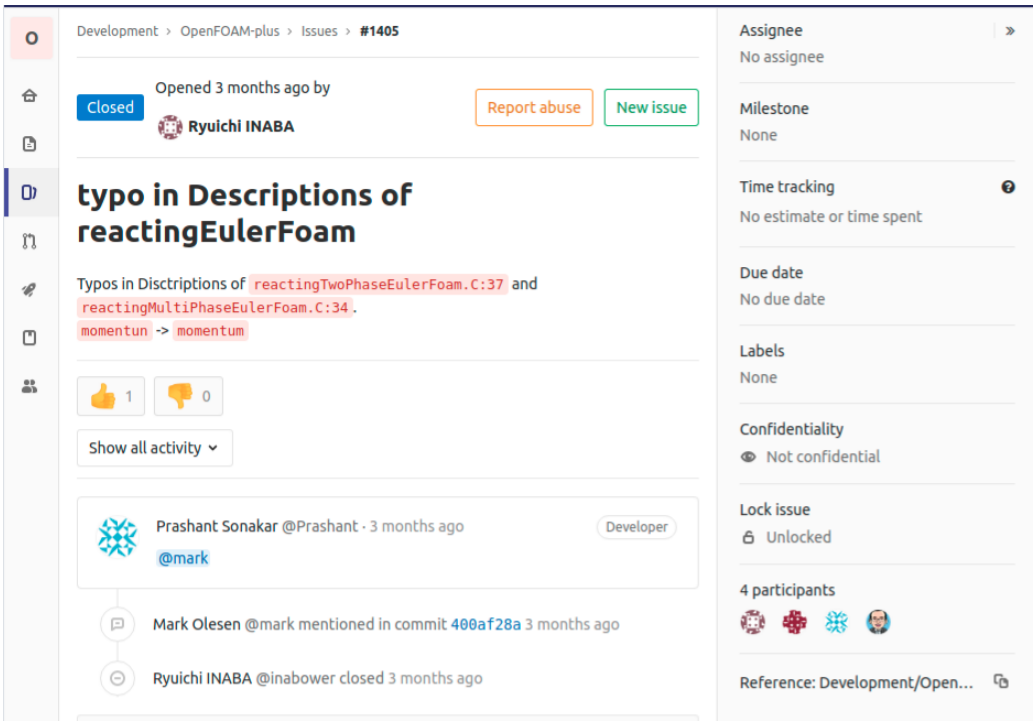


Fig. 2: An Example issue to fix typo.

しくは issue 内の議論へ参加することに限られる．なお 2019 年 11 月現在 OpenFOAM-plus の開発メンバーは 15 名であり，実力や面識，貢献頻度が担保されたメンバーで開発を行っていることが推察される．[?]

4. Foundation 版コントリビュート方法

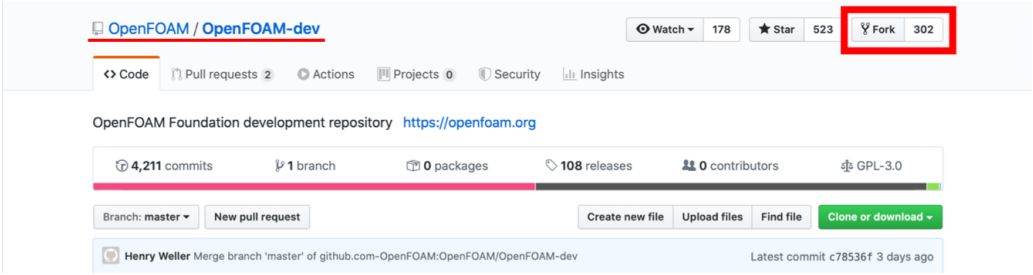
Foundation 版の開発は Github で行われている．Foundation 版では issue の管理は Github では行わないので注意が必要である．バグ報告は専用の issueTracking のページへ登録する必要がある．未登録の場合は，名前とメールアドレスを入力し，返信のリンクを辿ることで viewer として参加することができる．また数時間後にスパムでないことが認められると，reporter に昇格できる．reporter となることで issue を立てて，バグの報告が可能となる．ここでバグの重要度や改善方法を議論をすることになる．issueTracking の内容を確認すると，Foundation 版は基本的に bugfix を目的としたコントリビュートが主の様である．

重大なバグの修正または，新しい開発を提供する場合，OpenFOAM Foundation Contributor Agreement に署名する必要がある．Foundation 版 HP のメニューバーの Development/ContributorAgreeement のリンクを辿り，Contact Us のページから名前と所属，コントリビュートに参加した意思を記載して送信すると，後日，Henry G. Weller から PDF 形式の登録用紙が送付されてくる．登録用紙の内容は，前述の ContributorAgreeement のページに記載されているものと同じである．

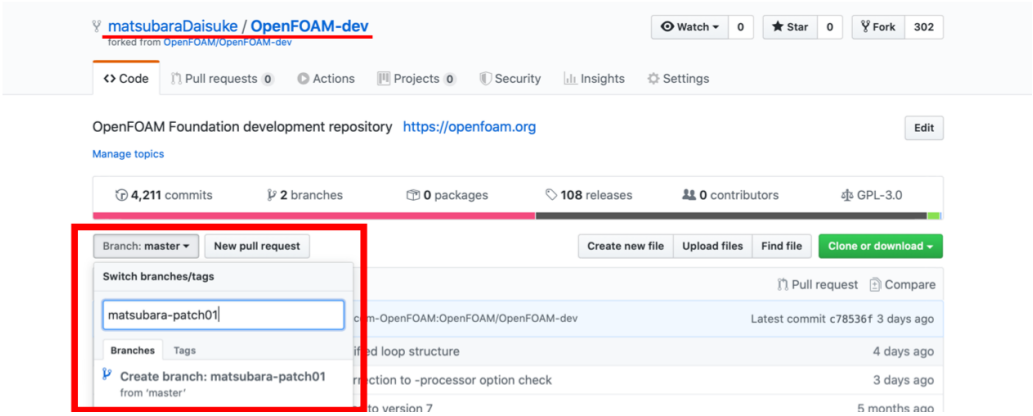
バグの修正などで Foundation 版の OpenFOAM の修正を開発元に共有する場合，Foundation 版の OpenFOAM は github 上で管理されているため，github アカウントが必要となる．

コントリビュートで最初に行うことは開発元のリポジトリを Fork することから始まる．(3(a)) Fork したこと

により，自身の github アカウントに OpenFOAM のリポジトリが作成されていることが確認できる．次に開発用の新しいブランチを作成する．ブランチ名は「(開発者名)-patch***」などとするとよいだろう．今回の例では「matsubara-patch1」とした．(3(b))



(a) A fork button of Github page



(b) Creating new branch

Fig. 3: A Explication to fork OpenFOAM-dev repository and to create your new branch.

次に個人の開発環境に移りターミナル上で git clone を行う．(図 4) 例：git clone https://github.com/(github アカウント名)/OpenFOAM-dev.git これによって，git clone したディレクトリ下に OpenFOAM のプロジェクトがコピーされる．

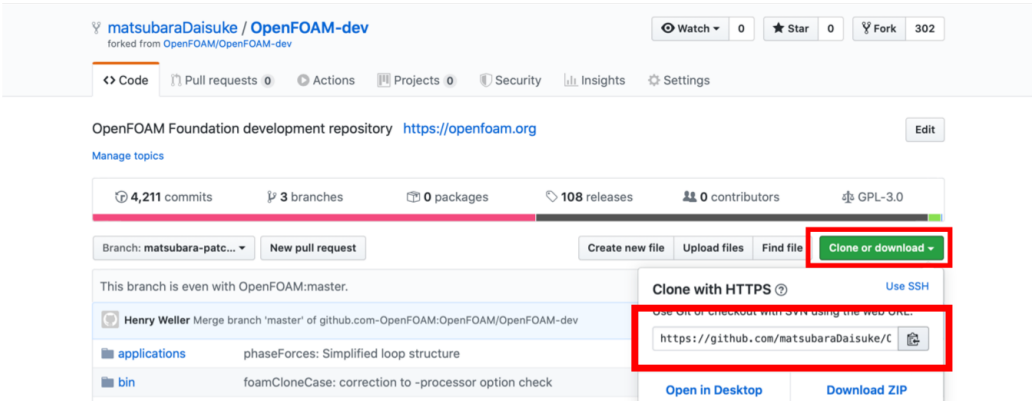


Fig. 4: Cloning repository on your local machine.

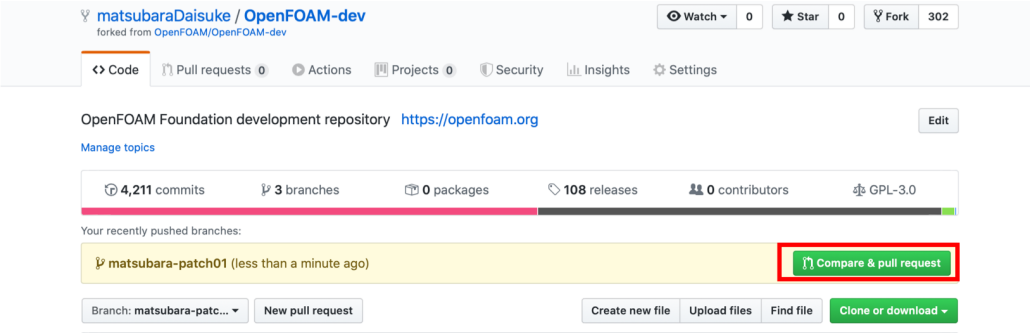
次に git checkout にて作成したブランチに切り替える 例：git checkout (開発者名)-patch*** OpenFOAM のプロジェクト内で修正や開発が完了した後，git add . git commit -m ” 変更内容のコメント” git push origin (開発者名)-patch*** とすることで，コードの修正内容を個人の github のリポジトリに反映させる．今回の例では以下の様な流れになる．

```
user@linux ~$  git clone https://github.com/matsubaraDaisuke/OpenFOAM-dev.git
user@linux ~$  cd OpenFOAM-dev
user@linux ~$  git checkout matsubara-patch1
```

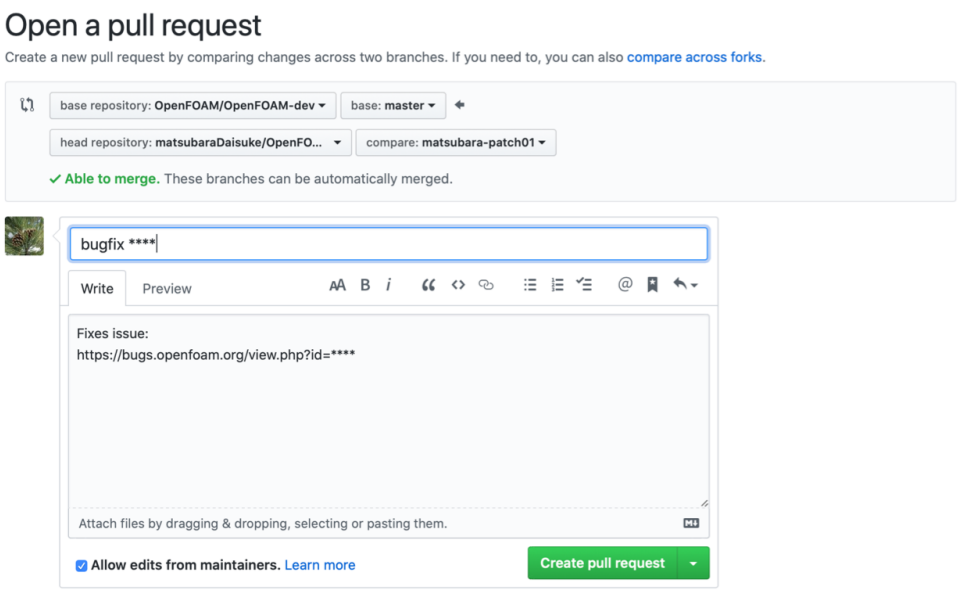
（コーディング）

```
user@linux ~$  git add .
user@linux ~$  git commit -m "bugfix ******"
user@linux ~$  git push origin matsubara-patch1
```

最後に github 上で pull request を開発元に対して行う．(5(a)) この時コメントに Fixes issue: https://bugs.openfoam.org/view.php?id=**** の様に最初に議論した issue の URL を添付するのが慣例のようである．(5(b))



(a) Pull request your branch into the master branch.



(b) Add your message to pull request.

Fig. 5: A explication to pull request your branch into the master branch.

5. Git の使い方

本節では https://github.com/OpenFOAM-jp/OpenFOAM-jp.git のリポジトリにコントリビュートの練習をする方法について解説する．環境は Ubuntu18.04 の環境を想定する．事前に GitHub のアカウントを作成する必要がある．コントリビュートをする際にはまず，Issue を立て自分が加えたい変更について議論する．Issue を立

てる際には以下の内容が含まれていることが望ましい．

- 機能の簡単な説明．問題が何であるかの明確で簡潔な説明．
- 希望するソリューションの説明．あなたが何をしたいのかについての明確で簡潔な説明．
- 検討した代替案を説明．検討した代替ソリューションまたは機能の明確で簡潔な説明．
- 追加のコンテキスト．機能リクエストに関する他のコンテキストまたはスクリーンショットをここに追加します．

OSS のコントリビュートのバージョン管理ソフトには Git が一般的に使用されている．OpenFOAM の開発においても Git が使用されている．まずは，Git をインストールする．

```
01 user@linux ~$  sudo apt install git
```

必要なのはソースだけで貢献しないのであれば，次のコマンドで clone することで作業は終わる．

```
01 user@linux ~$  git clone https://github.com/OpenFOAM-jp/OpenFOAM-jp.git
```

貢献をしたい場合は，OpenFOAM-jp のリポジトリを自分のアカウントに Fork する．
TODO: Fork の際の画面キャプチャを挿入する．
Fork 後は自分のアカウントのリポジトリを clone する．

```
01 user@linux ~$  git clone https://github.com/your_account_name/OpenFOAM-jp.git
```

clone をしたら自分の環境で実行環境を構築する．まずは，OpenFOAM を動かすために必要な外部ライブラリをインストールする．

```
01 user@linux ~$  sudo apt update
02 user@linux ~$  sudo apt upgrade
03 user@linux ~$  sudo apt install git-core
04 user@linux ~$  sudo apt install build-essential
05 user@linux ~$  sudo apt install cmake
06 user@linux ~$  sudo apt install libfl-dev
07 user@linux ~$  sudo apt install bison
08 user@linux ~$  sudo apt install zlib1g-dev
09 user@linux ~$  sudo apt install qt4-dev-tools
10 user@linux ~$  sudo apt install libqt4-dev
11 user@linux ~$  sudo apt install libqtwebkit-dev
12 user@linux ~$  sudo apt install gnuplot
13 user@linux ~$  sudo apt install libreadline-dev
14 user@linux ~$  sudo apt install libncurses-dev
15 user@linux ~$  sudo apt install libxt-dev
16 user@linux ~$  sudo apt install libopenmpi-dev
17 user@linux ~$  sudo apt install openmpi-bin
18 user@linux ~$  sudo apt install libboost-system-dev
19 user@linux ~$  sudo apt install libboost-thread-dev
20 user@linux ~$  sudo apt install libgmp-dev
21 user@linux ~$  sudo apt install libmpfr-dev
22 user@linux ~$  sudo apt install python
23 user@linux ~$  sudo apt install python-dev
24 user@linux ~$  sudo apt install libcgal-dev
25 user@linux ~$  sudo apt install curl
```

```
user@linux ~$  sudo apt install git-core
user@linux ~$  sudo apt install build-essential
user@linux ~$  sudo apt install cmake
user@linux ~$  sudo apt install libfl-dev
user@linux ~$  sudo apt install bison
user@linux ~$  sudo apt install zlib1g-dev
user@linux ~$  sudo apt install qt4-dev-tools
user@linux ~$  sudo apt install libqt4-dev
user@linux ~$  sudo apt install libqtwebkit-dev
user@linux ~$  sudo apt install gnuplot
user@linux ~$  sudo apt install libreadline-dev
user@linux ~$  sudo apt install libncurses-dev
user@linux ~$  sudo apt install libxt-dev
user@linux ~$  sudo apt install libopenmpi-dev
user@linux ~$  sudo apt install openmpi-bin
user@linux ~$  sudo apt install libboost-system-dev
user@linux ~$  sudo apt install libboost-thread-dev
user@linux ~$  sudo apt install libgmp-dev
user@linux ~$  sudo apt install libmpfr-dev
user@linux ~$  sudo apt install python
user@linux ~$  sudo apt install python-dev
user@linux ~$  sudo apt install libcgal-dev
user@linux ~$  sudo apt install curl
```

次に，OpenFOAM の bashrc をターミナル起動時に読み込むように /.bashrc に登録する．

```
user@linux ~$  echo "source  /OpenFOAM/OpenFOAM-dev/etc/bashrc" >>  /.bashrc
user@linux ~$  .  /.bashrc
```

ThirdParty の中の ParaView をインストールする．この際に python を登録すると ParaView の python 拡張機能が使用可能になる．既に ParaView をインストールしている場合であっても，この ParaView は OpenFOAM の結果表にのプラグインなども同梱しているため，以下の手順でもう一つインストールすることを推奨する．

```
user@linux ~$  cd $WM_THIRD_PARTY_DIR
user@linux ~$  ./makeParaView -python -python-lib /usr/lib/x86_64-linux-gnu/libpython2.7.so.1.0
```

最後に OpenFOAM をインストールする．

```
user@linux ~$  wmRefresh
user@linux ~$  cd $WM_PROJECT_DIR
user@linux ~$  ./Allwmake -j | tee log.Allwmake
```

ソルバーを動かしてインストールの確認を行う．

```
user@linux ~$  mkdir $WM_PROJECT_USER_DIR
user@linux ~$  cd $WM_PROJECT_USER_DIR
user@linux ~$  cp -r $FOAM_TUTORIALS/basic/potentialFoam/pitzDaily/ ./
user@linux ~$  cd pitzDaily/
user@linux ~$  foamRunTutorials
user@linux ~$  paraFoam
```

これで ParaView が起動されて結果の表示ができたら完成である．テストが全てパスしたらソースを変更する．

master ブランチで直接変更することはできないため，ファイルを変更する前に開発ブランチを作成する．

```
01 user@linux ~$  git branch branch_name
```

```
01 user@linux ~$  git checkout branch_name
```

branch_name には任意の名前を入れる．自分のアカウント名と加えたい変更について言及されていると分かりやすい．最初のコマンドでブランチを作成し，2 番目のコマンドでブランチに移動する．これにより，変更を行う準備はほぼ完了である．変更のラベルを付けるために，連絡先の名前と電子メールを以下のコマンドで指定する．

```
01 user@linux ~$  git config --global user.name "Your Name Comes Here"
02 user@linux ~$  git config --global user.email you@yourdomain.example.com
```

もし src/toto.cc というファイルをいくつか変更したり，新しいファイルとして追加したら，ローカルのコミットは次のコマンドで行う．”Your extensive commit message here #1” には変更に関するメッセージを追加する．#1 の部分は自分が追加したイシューの番号とする．

```
01 user@linux ~$  git add src/toto.cc
02 user@linux ~$  git commit -m "Your extensive commit message here #1"
```

この段階ではコミットはあなたのローカルリポジトリで行われていますが，GitHub リポジトリでは行われない．十分なテストで変更を検証したら，以下のコマンドで GitHub の自分のアカウントのリポジトリに変更を移すことができる．

```
01 user@linux ~$  git push origin branch_name
```

TODO: コマンドのメッセージを含める
このコマンドのメッセージに図のような URL が表示される．URL にアクセスしプルリクエストを作成する．
TODO: GetFEM のドキュメントから引用を行っているため言及する．
GetFEM++ のマスターブランチにマージすることは許可されていないので，あなたの役割はここで終わる．プルリクエストのページで管理者や他の開発者と議論することができる．管理者が承認した場合には変更がマージされる．
いくつかの便利な git コマンドを示す．

```
01 user@linux ~$  git status : status of your repository / branch
02 user@linux ~$  git log --follow "filepath" : Show all the commits modifying the
   specified file (and follow the eventual change of name of the file).
03 user@linux ~$  gitk --follow filename : same as previous but with a graphical interface
```

6. GitHub および Travis による継続的インテグレーション

Travis CI は，GitHub 上のソフトウェアのビルドやテストを行う，オンラインで分散型の継続的インテグレーション (CI) サービスである．継続的インテグレーションサービスとは，ビルドやテストを継続的に自動実行するサービスである．継続的インテグレーションを行うことにより，機能追加やリファクタリングによるデグレードを防ぐことができる．以下に，GetFEM++ で使用している.travis.yml を示す．リポジトリに yml ファイルを追加することで，Travis によるインテグレーションテストが実行される．今後 OpenFOAM のリポジトリでも CI が実行できるようにする予定である．

```
1 language: python
2 python:
3   - "3.6"
```

```
sudo: false
dist: bionic
cache:
  directories:
  - $HOME/.cache/pip
before_install:
- sudo apt-get install -y --no-install-recommends automake
- sudo apt-get install -y --no-install-recommends libtool
- sudo apt-get install -y --no-install-recommends make
- sudo apt-get install -y --no-install-recommends g++
- sudo apt-get install -y --no-install-recommends libqd-dev
- sudo apt-get install -y --no-install-recommends libqhull-dev
- sudo apt-get install -y --no-install-recommends libmumps-seq-dev
- sudo apt-get install -y --no-install-recommends liblapack-dev
- sudo apt-get install -y --no-install-recommends libopenblas-dev
- sudo apt-get install -y --no-install-recommends libpython3-dev
- sudo apt-get install -y --no-install-recommends ufraw
- sudo apt-get install -y --no-install-recommends imagemagick
- sudo apt-get install -y --no-install-recommends fig2dev
- sudo apt-get install -y --no-install-recommends texlive
- sudo apt-get install -y --no-install-recommends xzdec
- sudo apt-get install -y --no-install-recommends fig2ps
- sudo apt-get install -y --no-install-recommends gv
- pip install -r requirements.txt
addons:
  apt:
    update: true
script:
- bash autogen.sh
- export CXXFLAGS=-coverage
- export LDFLAGS=-coverage
- export CPPFLAGS=-coverage
- export CFLAGS=-coverage
- export FCFLAGS=-coverage
- ./configure --with-pic
- make -j8
- make -j8 check
- (cd doc/sphinx; make html)
after_success:
- bash <(curl -s https://codecov.io/bash)
```

TODO Travis CI について Wikipedia からの引用であることを明記する．

TODO 継続的インテグレーションサービスについて Wikipedia からの引用であることを明記する．

7. OpeFOAM-jp

OpenFOAM-jp[7] は今回立ち上がったコミュニティの活動を行うための GiuHub レポジトリである．目的は日本の OpenFOAM コミュニティの活性化，OpenFOAM へのコントリビュートの入り口，および OSS コントリビュート活動の学習の場としている．参加は誰でも国籍を問わず可能であり, 今後人口が増えてオープンで有意義な場として機能することを期待する．

7.1. OpenFOAM-jp/issues

issues レポジトリ [8] は最初の議論を行う場として設置された．テキストエディタ Vim の日本コミュニティのレポジトリ [9] の Fork であり，運用方法もこれを参考にする．ここでは OpenFOAM の新機能の提案や OpenFOAM-jp で行う活動の議論などを行うことを目的としている．質問掲示板である Google Forum と役割が重複すると混乱や情報の偏りを発生させるため明確な差別化を行うように注意して運用していく．

7.2. OpenFOAM-jp/OpenFOAM-jp

OpenFOAM-jp レポジトリ [10] は Foundation 版の OpenFOAM-dev の Fork であり，push や Pull Request などの git の基本機能の練習用に設置された．branch を作成してオリジナルの機能を配布する目的で使用することも期待している．エラーメッセージの日本語化プロジェクトなども予定している．

7.3. OpenFOAM-jp/OpenFOAM-utilities-tutorials-jp

OpenFOAM-utilities-tutorials-jp[11] は，web 上にあまり情報のない OpenFOAM の utilities に関するまとめを行う目的で設置された．その一例を図 6に示す．各 utility について，使用方法を MarkDown 形式のドキュメ

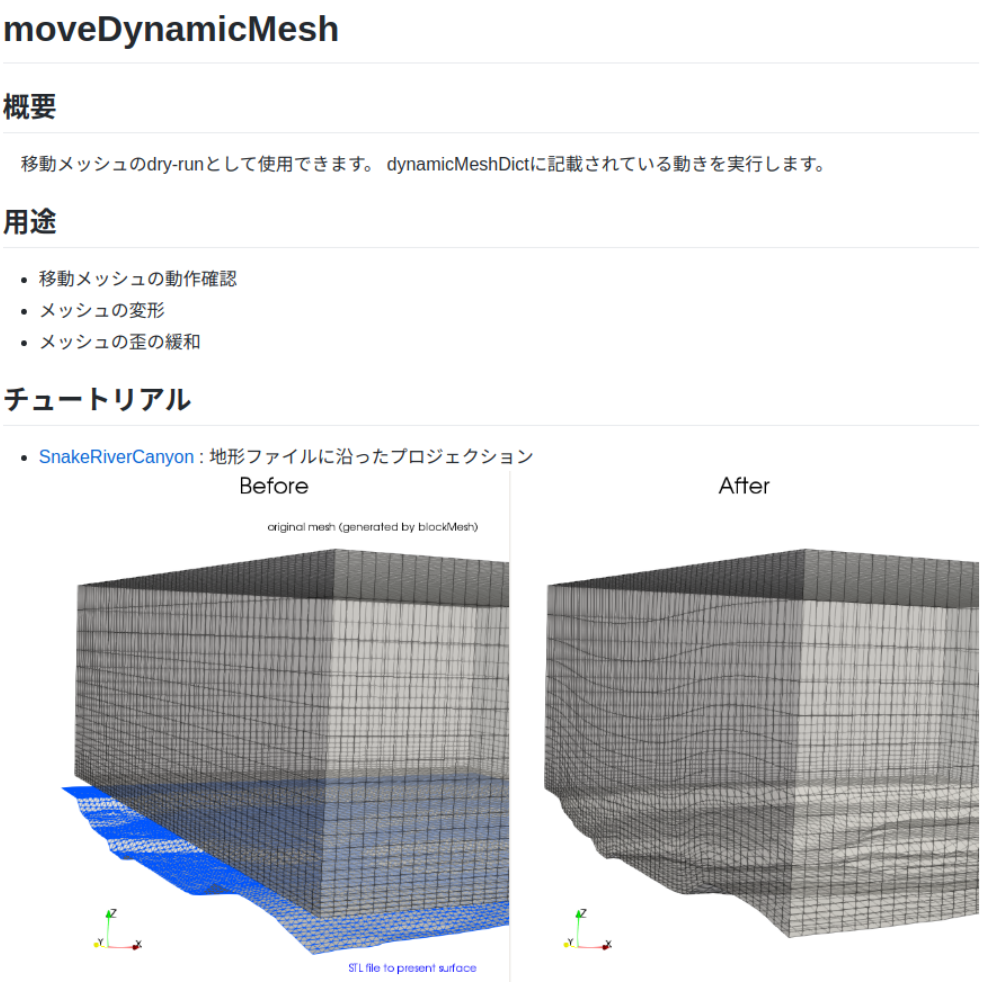


Fig. 6: moveDynamicMesh(v1906) utilitiy's tutorial document.[12]

ントでまとめ，そのチュートリアルを作成する．ディレクトリ構成は実際のソースコードに準拠し，最表層の MarkDown ファイルから各ドキュメントにアクセスできるようにしている．[13] 現時点では OpenFOAM-v1906 について作成中である．

8. まとめ

参考文献

[1] ESI Group, OpenCFD Inc. OpenFOAM History, https://www.openfoam.com/history/, (accessed 2019-11-24).

[2] 三邉 考志 OpenFOAM 開発・ファンディングプロジェクトの概要と事例 オープン CAE シンポジウム 2015 梗概集，GP23，2015. http://www.opencae.or.jp/wp-content/uploads/2016/01/OpenCAESympo-

sium2015_GP23_Abstract.pdf, (accessed 2019-11-24).
[3] OpenFOAM Foundation Inc. openfoam.org, https://openfoam.org/, (accessed 2019-11-24).
[4] OpenFOAM Foundation Inc. OpenFOAM-dev, https://github.com/OpenFOAM/OpenFOAM-dev, (accessed 2019-11-24).
[5] ESI Group, OpenCFD Inc. openfoam.com, https://openfoam.com/, (accessed 2019-11-24).
[6] ESI Group, OpenCFD Inc. OpenFOAM-plus, https://develop.openfoam.com/Development/OpenFOAM-plus, (accessed 2019-11-24).
[7] OpenFOAM-jp, https://github.com/OpenFOAM-jp, (accessed 2019-11-24).
[8] OpenFOAM-jp/issues, https://github.com/OpenFOAM-jp/issues, (accessed 2019-11-24).
[9] vim-jp, https://github.com/vim-jp, (accessed 2019-11-24).
[10] OpenFOAM-jp/OpenFOAM-jp, https://github.com/OpenFOAM-jp/OpenFOAM-jp, (accessed 2019-11-24).
[11] OpenFOAM-jp/OpenFOAM-utilities-tutorials-jp, https://github.com/OpenFOAM-jp/OpenFOAM-utilities-tutorials-jp, (accessed 2019-11-24).
[12] movedDynamicMesh example, https://github.com/OpenFOAM-jp/OpenFOAM-utilities-tutorials-jp/blob/master/v1906/mesh/manipulation/moveDynamicMesh/moveDynamicMesh.md, (accessed 2019-11-24).
[13] utilities tutorials index, https://github.com/OpenFOAM-jp/OpenFOAM-utilities-tutorials-jp/tree/master/v1906, (accessed 2019-11-24).