

## OpenFOAM コントリビュート活動

稲葉竜一<sup>1†</sup> 松原 大輔<sup>2</sup> @tkoyama010<sup>1</sup>

<sup>1</sup>OpenFOAM-jp <sup>2</sup> オープン CAE 勉強会

## OpenFOAM Contributing Activities

Ryuichi INABA<sup>\*†</sup> Daisuke MATSUBARA<sup>\*\*</sup> @tkoyama010<sup>\*</sup>

<sup>\*</sup>OpenFOAM-jp <sup>\*\*</sup>OpenCAE Local user group

### Abstract

The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog. The quick brown fox jumps over the lazy dog.

**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 を最新版とするフォークが”Foundation 版”[3][4] と呼ばれている。

一方で OpenFOAM には多くのフォークが存在する。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 からフォークした 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、このフォークが”ESI 版”[5][6] もしくは”plus 版”と呼ばれる。

本報告ではこの Foundation 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。[develop.openfoam.com/Development/OpenFOAM-plus](develop.openfoam.com/Development/OpenFOAM-plus) 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。

GitLab のページの右上のボタン [図 1] 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 中のスペルミス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

<sup>†</sup> E-mail address of corresponding author: inabower@gmail.co.jp



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

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

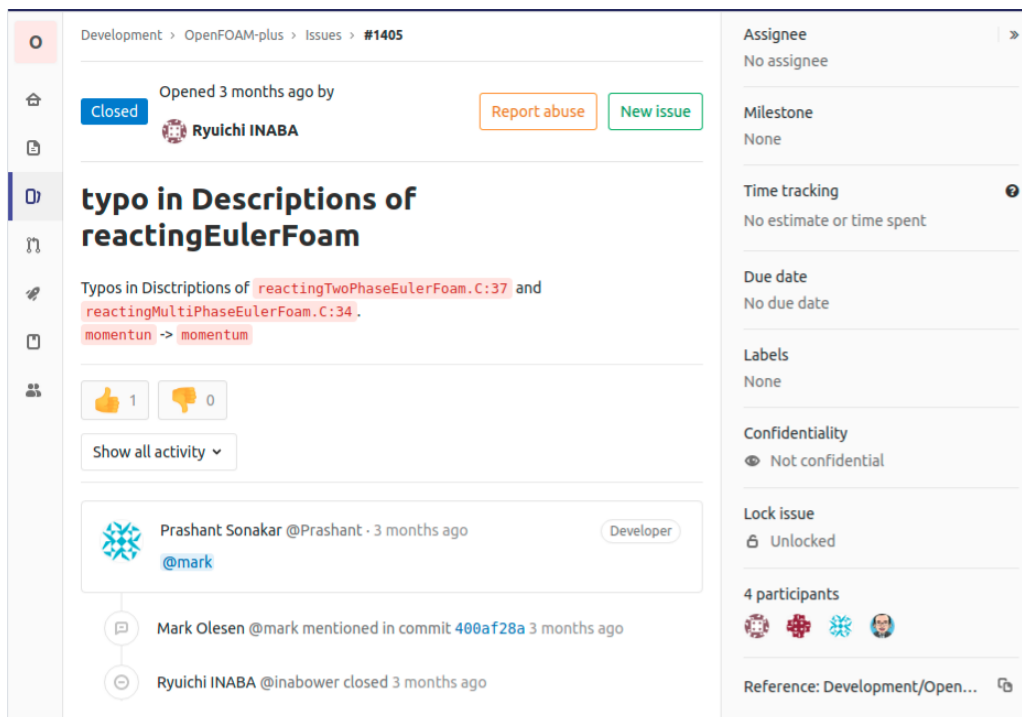


Fig. 2: An Example issue to fix typo.

錬していくこととなる。

一方で、push や PR 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 Repository は Github で公開されているが、issue の管理は Foundation の HP の issue Tracking のページで行われる。

メールを送る → Henry G. Weller から登録用紙 (THE OPENFOAM FOUNDATION CONTRIBUTOR AGREEMENT) が送られてくる → 同意できる場合はサインと共に返送 → reporter として登録される。

#### 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、次のコマンドでクローン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フォークする。

TODO: フォークの際の画面キャプチャを挿入する。

フォーク後は自分のアカウントのリポジトリをクローンする。

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

クローン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 libcglib-dev
25 user@linux ~$ sudo apt install curl
```

```

26 user@linux ~$ sudo apt install git-core
27 user@linux ~$ sudo apt install build-essential
28 user@linux ~$ sudo apt install cmake
29 user@linux ~$ sudo apt install libfl-dev
30 user@linux ~$ sudo apt install bison
31 user@linux ~$ sudo apt install zlib1g-dev
32 user@linux ~$ sudo apt install qt4-dev-tools
33 user@linux ~$ sudo apt install libqt4-dev
34 user@linux ~$ sudo apt install libqtwebkit-dev
35 user@linux ~$ sudo apt install gnuplot
36 user@linux ~$ sudo apt install libreadline-dev
37 user@linux ~$ sudo apt install libncurses-dev
38 user@linux ~$ sudo apt install libxt-dev
39 user@linux ~$ sudo apt install libopenmpi-dev
40 user@linux ~$ sudo apt install openmpi-bin
41 user@linux ~$ sudo apt install libboost-system-dev
42 user@linux ~$ sudo apt install libboost-thread-dev
43 user@linux ~$ sudo apt install libgmp-dev
44 user@linux ~$ sudo apt install libmpfr-dev
45 user@linux ~$ sudo apt install python
46 user@linux ~$ sudo apt install python-dev
47 user@linux ~$ sudo apt install libcglib-dev
48 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。

```

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

```

ThirdParty 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。

```

01 user@linux ~$ cd $WM_THIRD_PARTY_DIR
02 user@linux ~$ ./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。

```

01 user@linux ~$ wmRefresh
02 user@linux ~$ cd $WM_PROJECT_DIR
03 user@linux ~$ ./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。

```

01 user@linux ~$ mkdir $WM_PROJECT_USER_DIR
02 user@linux ~$ cd $WM_PROJECT_USER_DIR
03 user@linux ~$ cp -r $FOAM_TUTORIALS/basic/potentialFoam/pitzDaily/ ./
04 user@linux ~$ cd pitzDaily/
05 user@linux ~$ foamRunTutorials
06 user@linux ~$ paraFoam

```

これで ParaView が起動されて結果の表示ができれば完成である。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 이슈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ません。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 yaml 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"
```

```

4  sudo: false
5  dist: bionic
6  cache:
7    directories:
8      - $HOME/.cache/pip
9  before_install:
10 - sudo apt-get install -y --no-install-recommends automake
11 - sudo apt-get install -y --no-install-recommends libtool
12 - sudo apt-get install -y --no-install-recommends make
13 - sudo apt-get install -y --no-install-recommends g++
14 - sudo apt-get install -y --no-install-recommends libqtd-dev
15 - sudo apt-get install -y --no-install-recommends libqhull-dev
16 - sudo apt-get install -y --no-install-recommends libmumps-seq-dev
17 - sudo apt-get install -y --no-install-recommends liblapack-dev
18 - sudo apt-get install -y --no-install-recommends libopenblas-dev
19 - sudo apt-get install -y --no-install-recommends libpython3-dev
20 - sudo apt-get install -y --no-install-recommends ufw
21 - sudo apt-get install -y --no-install-recommends imagemagick
22 - sudo apt-get install -y --no-install-recommends fig2dev
23 - sudo apt-get install -y --no-install-recommends texlive
24 - sudo apt-get install -y --no-install-recommends xzdec
25 - sudo apt-get install -y --no-install-recommends fig2ps
26 - sudo apt-get install -y --no-install-recommends gv
27 - pip install -r requirements.txt
28 addons:
29   apt:
30     update: true
31   script:
32     - bash autogen.sh
33     - export CXXFLAGS=-coverage
34     - export LDFLAGS=-coverage
35     - export CPPFLAGS=-coverage
36     - export CFLAGS=-coverage
37     - export FCFLAGS=-coverage
38     - ./configure --with-pic
39     - make -j8
40     - make -j8 check
41     - (cd doc/sphinx; make html)
42   after_success:
43     - 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. まとめ

### 参考文献

- [1] ESI Group, OpenCFD Inc. OpenFOAM History, <https://www.openfoam.com/history/>, (accessed 2019-11-24).
- [2] 三邊 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/OpenCAESymposium2015\\_GP23\\_Abstract.pdf](http://www.opencae.or.jp/wp-content/uploads/2016/01/OpenCAESymposium2015_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).