

## 第11回オープンCAE初心者勉強会

# SalomeMecaとOpenFOAMの連携(メッシュ変換)及び これをスムーズに行う為のTreeFoamの紹介

# SalomeMecaでメッシュ(unv形式)を切り、Foam形式に変換

Salome側で定義したface、volumeのグループがそのグループ名ごとFoam形式に変換できるものを作成。

(patch、faceSet、cellSet、faceZone、cellZoneが名前を引き継いで作成する。)

## <作成した動機>

### <OpenFOAMで複数の物性値を定義する方法(08/9月当時)>

1. モデル全体のメッシュを切る。
2. メッシュモデル全体から定義したい部分(領域をstl形式で定義)を抜き出す。  
領域をstl形式で保存、cellSetDictを作り、cellsetを実行
3. 抜き出した部分にデータをセット  
setFieldsDictを作り、setFieldsを実行
4. 結果を確認  
paraFoamでセットした値を確認

### <デメリット>

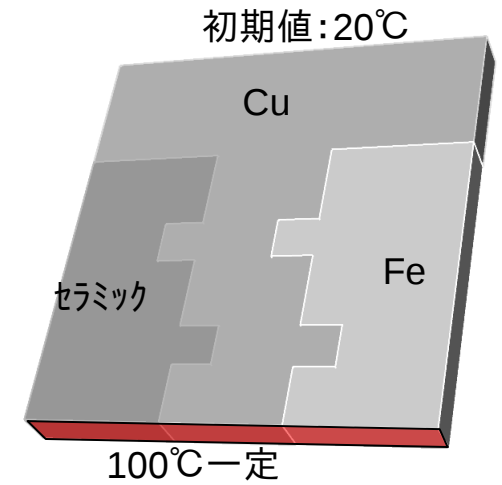
- ・定義したい領域のモデルを作り、cellsetDictを作る必要がある。
- ・既にメッシュがある状態から、部分的にメッシュを取り出すので、境界が正確に抜き出せない。
- ・全て、文字ベースで実行される為、領域名とモデルにイメージがつかみ難い。

具体例:各領域の境界を確認した結果(08/9月当時:OF-1.4.1)

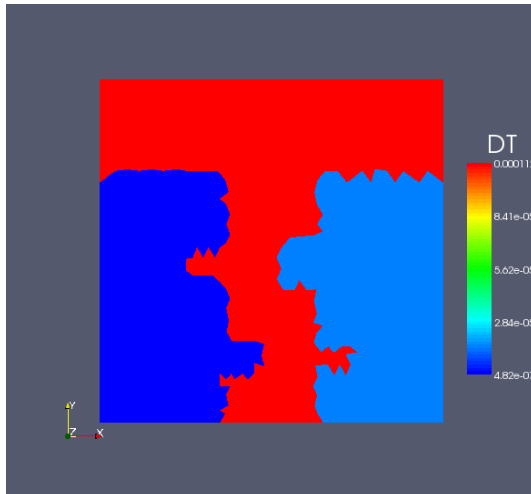
モデルを領域分割して、熱流束解析を実施。

<物性値>

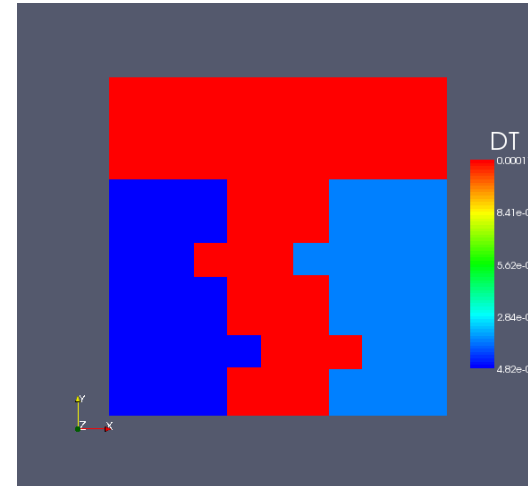
項目	記号	単位	Cu	Fe	セラミック	備考
密度	$\rho$	kg/m <sup>3</sup>	8.96e3	7.83e3	2.30e3	
比熱	C	J/kg.K	383	465	1046	
熱伝導率	$\lambda$	J/m.s.K	386	52.8	1.162	
熱拡散係数	$\alpha$	m <sup>2</sup> /s	112e-6	14.5e-6	0.482e-6	$\lambda/\rho C$



cellSetで領域を抜き出して  
値を設定



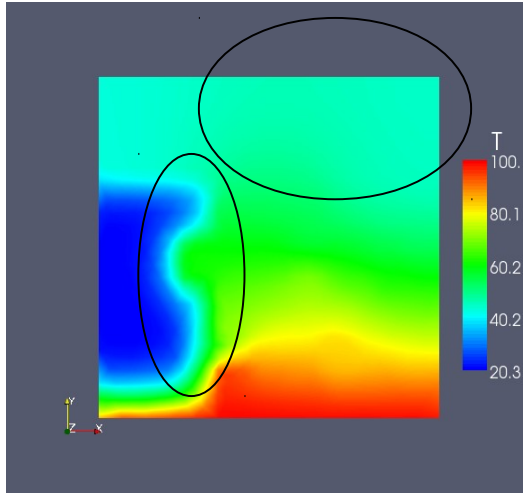
salomeMeca側で領域を定義し  
値を設定



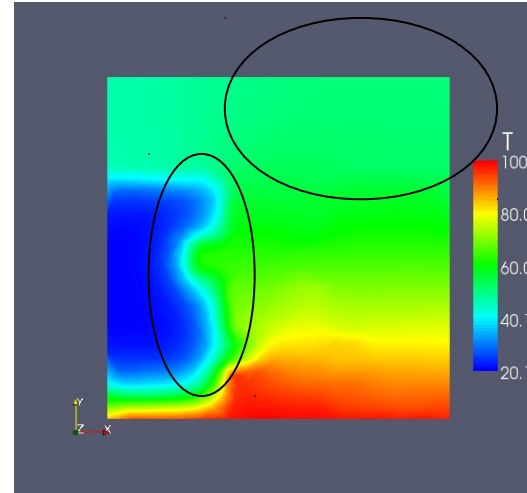
境界が正確に定義できている

## 具体例:計算結果(60s後の結果)

cellSetで抜き出し



SalomeMeca



結果が少し異なっている

SalomeMecaでメッシュを作り、変換すると領域の境界がきれいに切れる。  
この手順を説明。

# 1. SalomeMecaで作ったメッシュを変換する方法

damBreakを例にとる

## 1-1. Caseの作成

tutorialsをコピーして、case「damBreak-test」を作成。

(OpenFOAM/xxx-1.7.1/run/tutorials/multiphase/interFoam/laminar/damBreak)

caseフォルダ内にmodelフォルダを作成する。

test

damBreak-test      caseフォルダ

0

constant

model

system

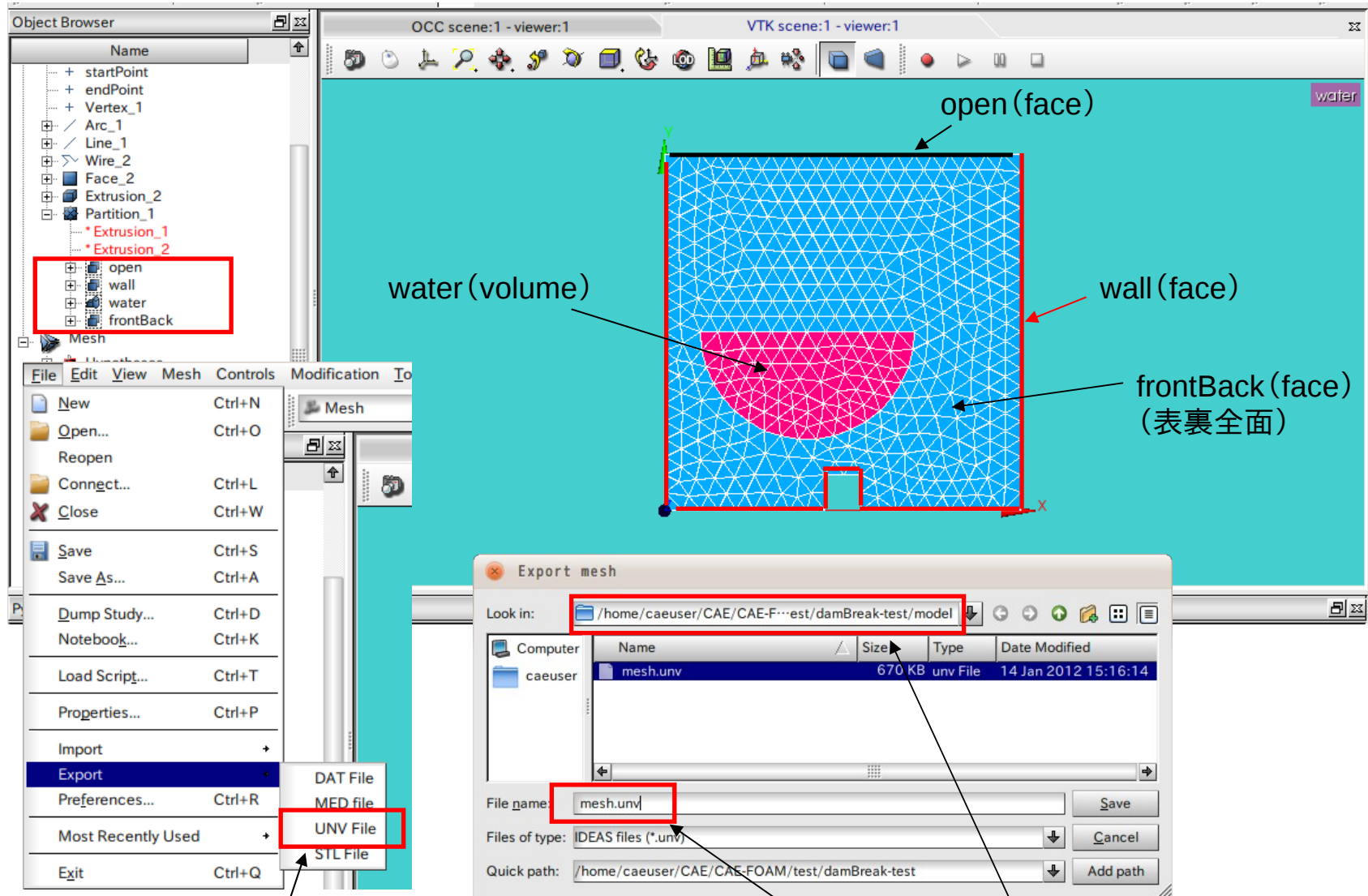
新たに作成(この中にSalomeMeca関係のファイルを置く)

modelフォルダ内にSalomeMecaで作成したunv形式のメッシュファイルを置き

このメッシュファイル(mesh.unv)をFoam形式に変換する。

(変換するファイルを固定して、後処理が楽になるようにした)

## 1-2. SalomeMeca側でメッシュ作成



UNV形式を選択

ファイル名を「mesh.unv」として  
damBreak-test/modelフォルダ内に保存する

## 1-3. メッシュ変換 (unv形式 → Foam形式)

FOAM端末を起動し「unv2gmshToFoam」を実行

赤字を新たに作成した

```

caeuser@ubuntu1010V: ~/CAE/CAE-FOAM/test/damBreak-test
ファイル(F) 編集(E) 表示(V) 検索(S) 端末(T) ヘルプ(H)

OpenFOAM-1.7.1
--FOAM端末を起動しました。

caeuser@ubuntu1010V:~/CAE/CAE-FOAM/test/damBreak-test$ unv2gmshToFoam

Writing zone 0 to cellZone cellZone_0 and cellSet
Writing zone 1 to cellZone cellZone_1 and cellSet
Writing zone 0 to faceZone faceZone_0 and faceSet
End

unv2gmshToFoam メッシュ変換は、完了しました。

<unv2gmshToFoamSmooth>を実行します。

gmshToFoamファイルを整形中...
unv2gmshファイを整形中...
Salome側 (unv file) のグループ名を取得...
Foam側で設定したpatch名を取得...
定義されていないcellZoneがあります。cellZone_defaultで定義しました
Salome側 (unv file) のグループ名を取得...
boundaryファイルに設定されたpatch名をSalome側の名称に変換...
faceZones, cellZones, sets内ファイル名をsalome側名称に変換...
Salomeのグループ名をFoam側の名称に変換しました。

caeuser@ubuntu1010V:~/CAE/CAE-FOAM/test/damBreak-test$

```

## &lt;処理内容&gt;

「model/mesh.unv」ファイルを  
 unv → gmsh形式に変換  
 (unv2gmshを実行)  
 gmsh → Foam形式に変換  
 (gmshToFoamを実行)  
 Salomeのグループ名をFoam側に設定  
 (unv2gmshToFoamSmoothを実行)

(ideasUnvToFoamもあるが、  
 これは、volumeを認識しないので  
 unv2gmshToFoamを作成した)

これにより、salomeで設定した  
 グループ名がそのまま  
 patch、  
 faceZone、cellZone、  
 faceSet、cellSet  
 の名称に適用される。

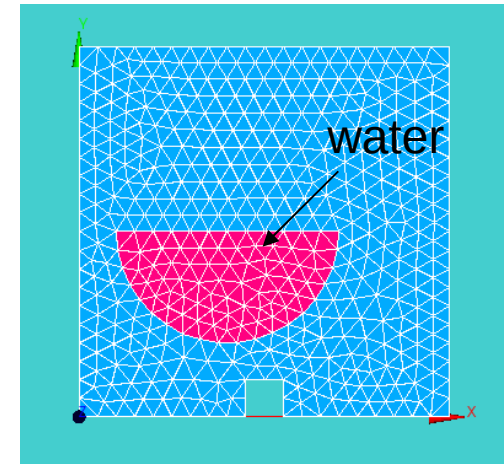
## 1-4. メッシュ変換後のboundaryとsetsフォルダの内容

## Boundaryの内容

変換前 (tutorial)	変換後 (SalomeMeca)
<pre> 5 (   leftWall   {     type      wall;     nFaces   50;     startFace 4432;   }   rightWall   {     type      wall;     nFaces   50;     startFace 4482;   }   lowerWall   {     type      wall;     nFaces   62;     startFace 4532;   }   atmosphere   {     type      patch;     nFaces   46;     startFace 4594;   }   defaultFaces   {     type      empty;     nFaces   4536;     startFace 4640;   } ) </pre>	<pre> 5 (   Default   {     type      patch;     nFaces   0;     startFace 4963;   }   frontBack   {     type      patch;     nFaces   1712;     startFace 4963;   }   wall   {     type      patch;     nFaces   248;     startFace 6675;   }   open   {     type      patch;     nFaces   78;     startFace 6923;   }   defaultFaces   {     type      patch;     nFaces   0;     startFace 7001;   } ) </pre>

Patch名がSalomeグループ名に変わる

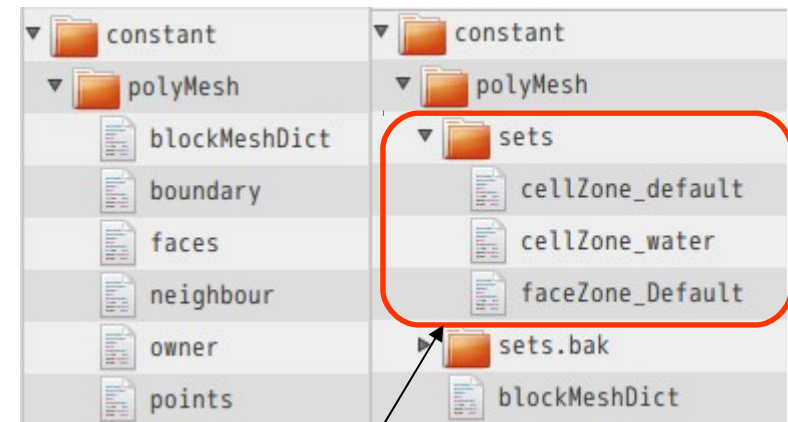
タブの幅:



## polyMeshの内容

&lt;変換前&gt;

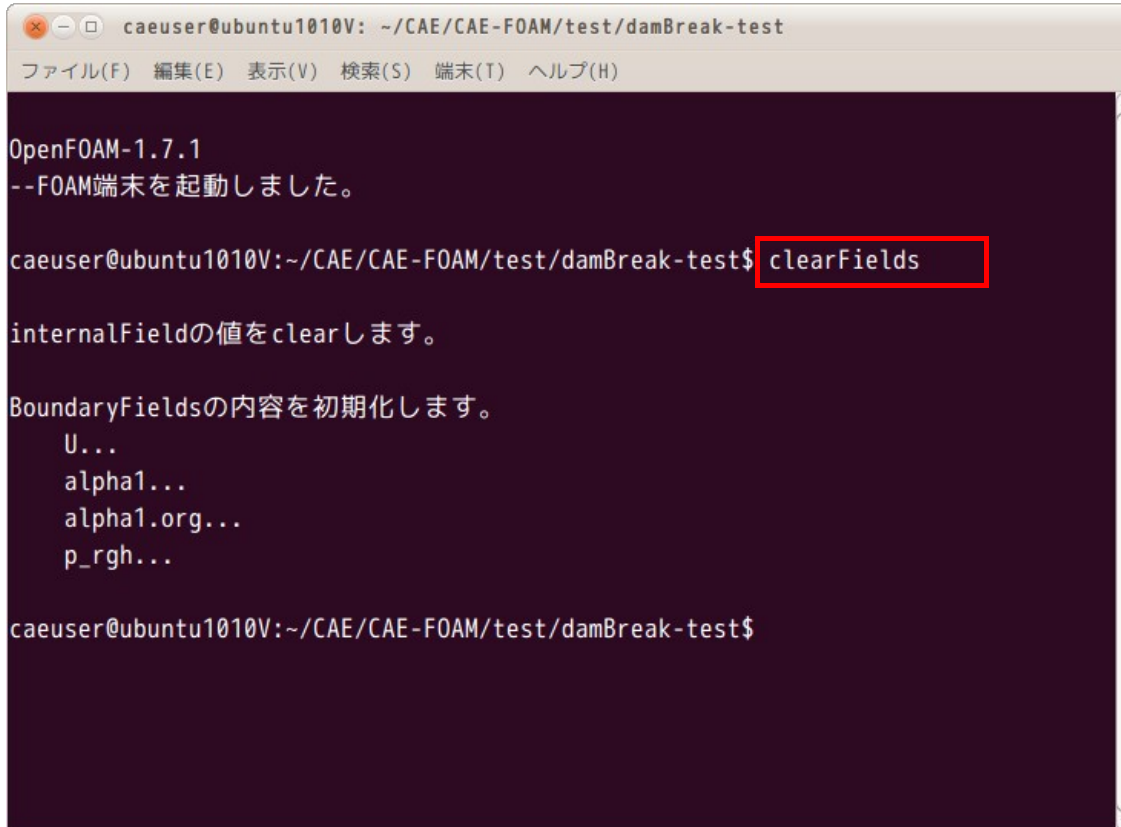
&lt;変換後&gt;

setsフォルダが追加され  
cellZone\_waterができあがる



## 1-5. boundaryFieldのクリア

まだ、各fieldのboundaryFieldが変更されていないので、boundaryFieldの内容をboundaryのpatch名に合わせ、zerogradientにセットする。  
(FOAM端末を起動し「clearFields」を実行)

A terminal window titled 'caeuser@ubuntu1010V: ~/CAE/CAE-FOAM/test/damBreak-test'. The window shows the execution of 'OpenFOAM-1.7.1' and the command 'clearFields'. The output indicates that internal fields and boundary fields are being cleared. The command 'clearFields' is highlighted with a red box.

```
caeuser@ubuntu1010V: ~/CAE/CAE-FOAM/test/damBreak-test
OpenFOAM-1.7.1
--FOAM端末を起動しました。

caeuser@ubuntu1010V:~/CAE/CAE-FOAM/test/damBreak-test$ clearFields

internalFieldの値をclearします。

BoundaryFieldsの内容を初期化します。
U...
alpha1...
alpha1.org...
p_rgh...

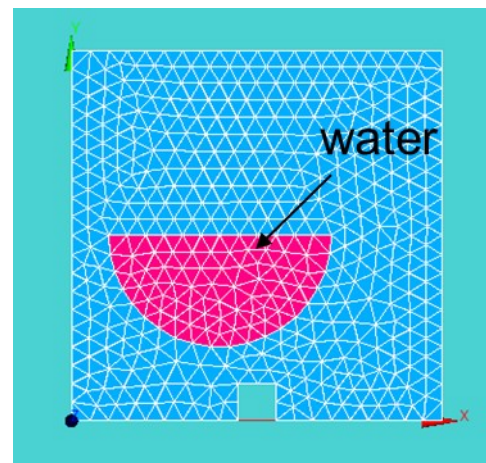
caeuser@ubuntu1010V:~/CAE/CAE-FOAM/test/damBreak-test$
```

この操作で全て設定が完了した事になり、形状をparaFoamで確認できる。

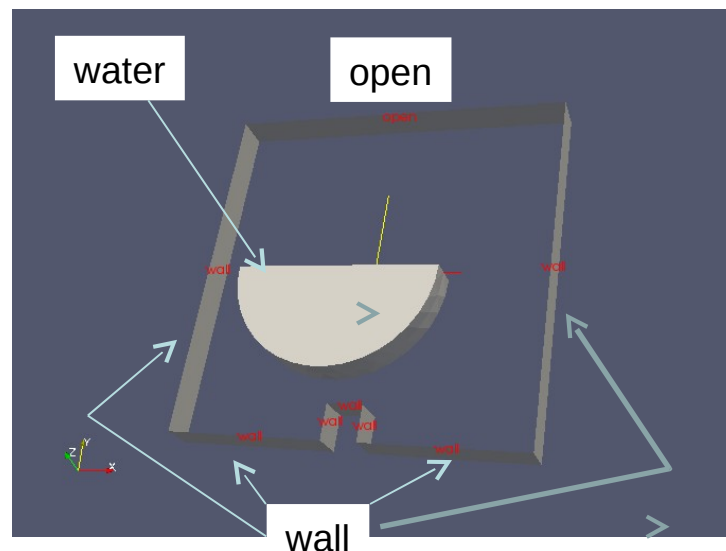
## 1-6. メッシュ変換結果

変換後のboundaryの内容

```
// ***** //  
5  
(  
  Default  
  {  
    type      patch;  
    nFaces    0;  
    startFace 4963;  
  }  
  frontBack  
  {  
    type      patch;  
    nFaces    1712;  
    startFace 4963;  
  }  
  wall  
  {  
    type      patch;  
    nFaces    248;  
    startFace 6675;  
  }  
  open  
  {  
    type      patch;  
    nFaces    78;  
    startFace 6923;  
  }  
  defaultFaces  
  {  
    type      patch;  
    nFaces    0;  
    startFace 7001;  
  }  
)  
// ***** //
```



## paraFoamによる確認結果



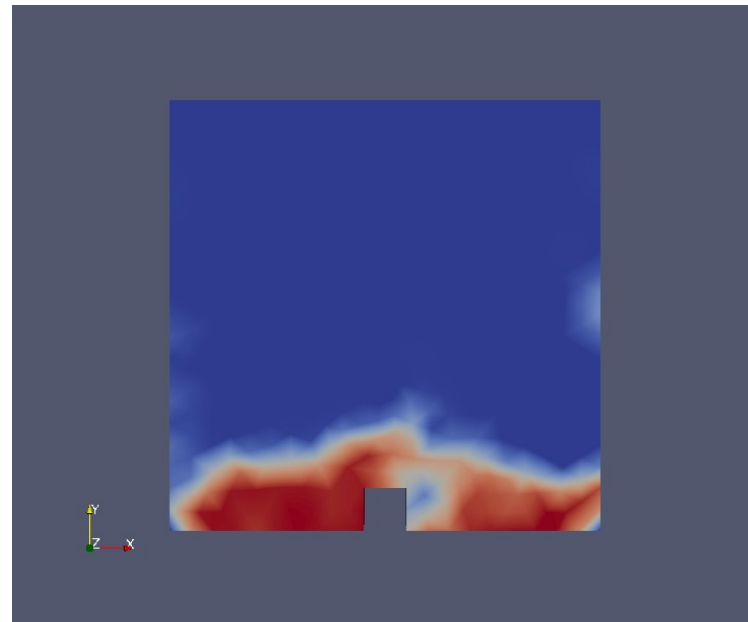
正常に変換ができています。

## 1-7. 境界条件、初期値の設定、実行

メッシュ変換ができたので、以下を設定してsolverを実行する。  
(通常通りに実行)

1. setFieldsで「alpha1」fieldの水の領域 (cellZone\_water) に「1.0」をセット
2. 各fieldのboundaryFieldを作成
3. interFoamを実行

<実行結果>



## 2. TreeFoamの紹介

SalomeMecaで作成したメッシュ(mesh.unv)変換する操作をスムーズに行う為、及びOpenFOAMの操作性を向上させる為、TreeFoamを作成した。

### 2-1. 今までの方法

#### OF-Launcher (従来)

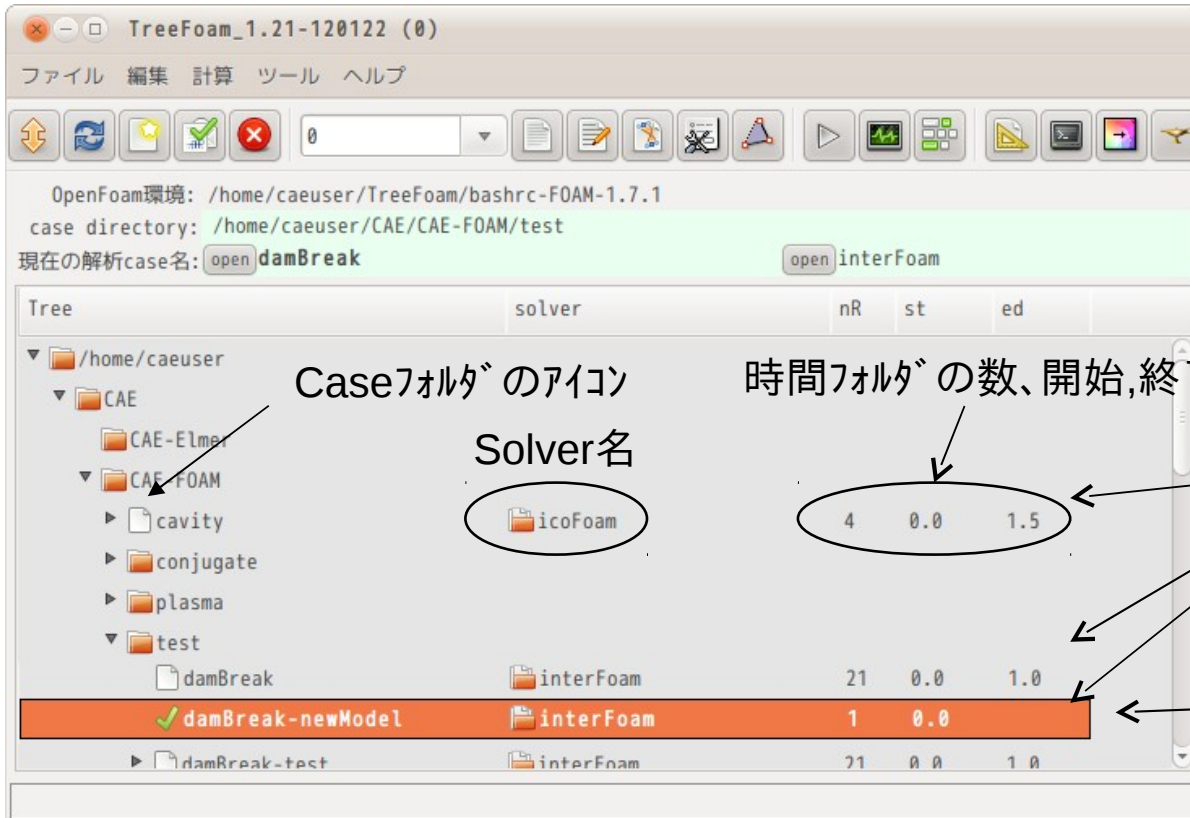


1. 今どこにいるか判り難い。  
(カレントディレクトリが判り難い)
2. ページをめくる感覚なので、必ず2アクションの操作が必要。
3. 境界条件の設定をスムーズに行いたい。  
fieldやpatchが多い場合、設定やその確認に時間が掛かるし、入力ミスも発生する。

TreeFoamを作成  
(gridEditor)

## 2-2. TreeFoamの概要

フォルダをツリーで表示。Caseフォルダを認識。



ツリーでフォルダを表示:カレントディレクトリが一目で判る

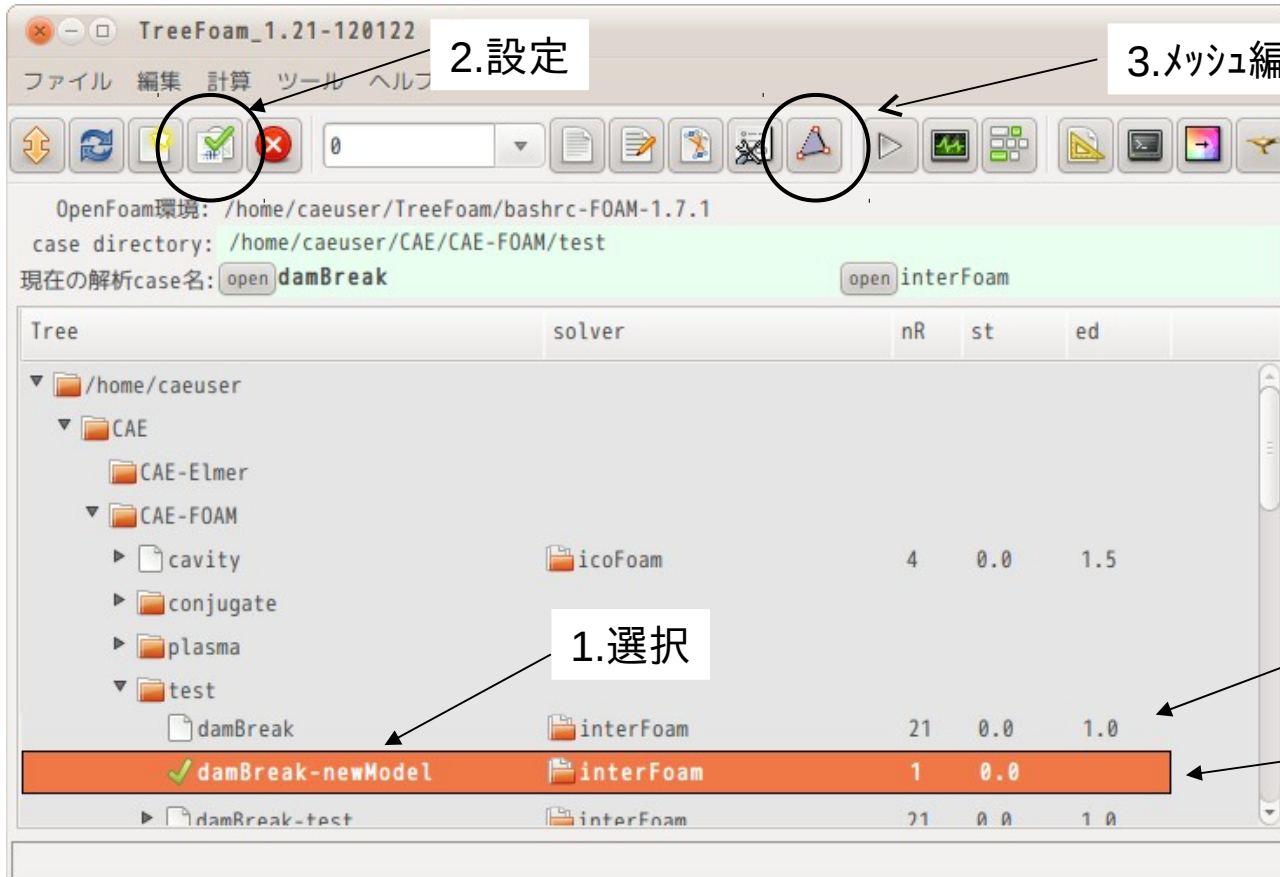
(OpenFOAMの解析case、使用solver、どこまで計算したかも判る様に表示)

一画面に全てのボタンを配置(1アクションで操作)

gridEditorを使って、境界条件をExcelの様に表形式(field×patch)で編集できる。

## 2-3. メッシュ変換の例

既にメッシュファイル(model/mesh.unv)が存在しているものとする



### 2-3. メッシュ編集(メッシュ変換、スケール設定)



1. クリック(volume)をグループ化している場合 (unv2gmshToFoamを実行)



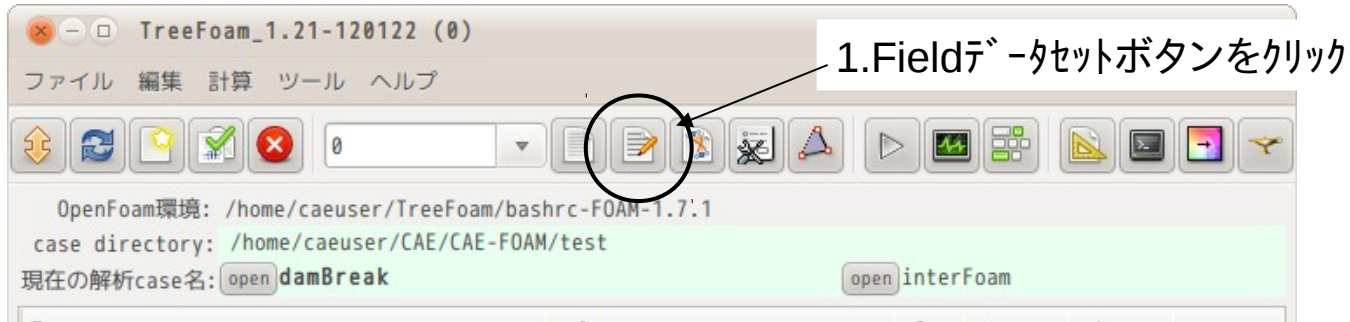
確認メッセージ



スケールの確認

メッシュ変換は完了

## 2-3. boundaryFieldのクリア



2.クリックして全ての「boundaryField」をクリアする。  
 (patch名をboundary⇔boundaryField間で合わせる)  
 internalFieldは、メッシュ変換時にクリアされる。

setFieldsコマンドでデータをfieldにセットする時、  
 internalFieldやboundaryFieldに矛盾があると  
 エラが発生する為、これらを合わせる為に  
 internalFieldとboundaryFieldをクリアする。



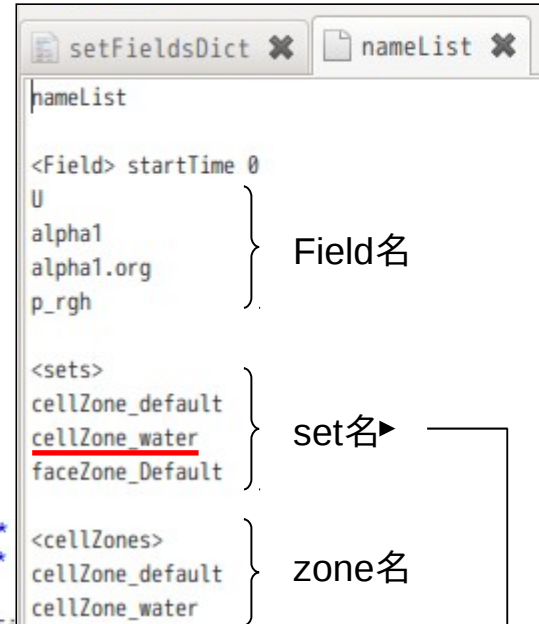
## 2-4. alpha1 Fieldにデータをセット

モデル(alpha1 Field)にwater領域を定義する。

<Gedit開いた内容(nameList)>

1. クリックしてsetFieldsDictを編集する。

- ・systemフォルダ内にある「setFieldsDict」と「nameList」(field名、set名、zone名のリスト)をgeditで開く
- ・setFieldsDictファイルがない場合はデフォルトのsetFieldsDictが開く



<setFieldsDict内容>

```
// *****
defaultFieldValues
(
  volScalarFieldValue alpha1 0
);
regions
(
  boxToCell
  {
    box (0 0 -1) (0.1461 0.292 1);
    fieldValues
    (
      volScalarFieldValue alpha1 1
    );
  }
);
// *****

// *****
defaultFieldValues
(
  volScalarFieldValue alpha1 0
);
regions
(
  cellToCell
  {
    set cellZone_water; cellSetName
    fieldValues
    (
      volScalarFieldValue alpha1 1
    );
  }
);
// *****
```

setFieldsDict保存後は、「setFields」ボタンをクリックして実行する



2. クリックしてsetFieldsコマンドを実行する。

## 2-5. 境界条件の設定 (gridEditorの起動)

境界条件の編集はgridEditorを起動して、ここで編集する。  
 (gridEditor:境界条件をExcelの様に表示形式で編集ができる)

**1. クリック**

**2. 「Gridで編集」ボタンをクリック gridEditorが起動する。**

**<gridEditor画面>**

**Field名**

setFieldsでデータセットした為 List形式になっている。

patch名

	define patch (boundary)	U	alpha1	alpha1.org	p_rgh
field type		volVectorField;	volScalarField;	volScalarField;	volScalarField;
dimensions		[0 1 -1 0 0 0];	[0 0 0 0 0 0];	[0 0 0 0 0 0];	[1 -1 -2 0 0 0];
internal Field		uniform (0 0 0);	nonuniform List<scalar> 2991 ( 0 0...	uniform 0;	uniform 0;
Default	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
frontBack	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
wall	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
open	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
defaultFaces	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;

## 2-6. 境界条件の編集 (gridEditorで編集)

複数起動したgridEditor間で「コピー」「貼付け」が可能なので、「damBreak」と「damBreak-newModel」の2つのgridEditorを起動する。

The image shows two overlapping windows of the gridEditor. The top window is titled 'Edit BoundaryFields: damBreak/0/. (0:1)' and the bottom window is 'Edit BoundaryFields: damBreak-newModel/0/. (0:0)'. Both windows display a table of boundary conditions with columns for 'define patch (boundary)', 'U', 'alpha1', 'alpha1.org', and 'p\_rgh'. The 'damBreak' window has rows for 'leftWall', 'rightWall', and 'lowerWall'. The 'damBreak-newModel' window has rows for 'wall', 'open', and 'defaultFac es'. Annotations with arrows point to specific elements: 'case名を表示' points to the window title; 'wallのデータを全て選択し、右クリックして「コピー」' points to the 'leftWall' row in the top window; 'wallのpatchTypeを選択し、右クリックして「貼付け」(画面は既に貼付けた状態)' points to the 'wall' row in the bottom window; and another arrow points to the 'wall' row in the bottom window.

	define patch (boundary)	U	alpha1	alpha1.org	p_rgh
field type dimensions	volVectorField; [0 1 -1 0 0 0 0];		volScalarField; [0 0 0 0 0 0 0];	volScalarField; [0 0 0 0 0 0 0];	volScalarField; [1 -1 -2 0 0 0 0];
internal Field	uniform (0 0 0);		nonuniform List<scalar> 2268 ( 1 1...	uniform 0;	uniform 0;
leftWall	type wall;	type fixedValue; value uniform (0 0 0);	type zeroGradient;	type zeroGradient;	type buoyantPressure; value uniform 0;
rightWall	type wall;	type fixedValue; value uniform (0 0 0);	type zeroGradient;	type zeroGradient;	type buoyantPressure; value uniform 0;
lowerWall	type wall;	type fixedValue; value uniform (0 0 0);	type zeroGradient;	type zeroGradient;	type buoyantPressure; value uniform 0;
atmosphere	type patch;				
defaultFac es	type empty;				

	define patch (boundary)	U	alpha1	alpha1.org	p_rgh
field type dimensions		volVectorField; [0 1 -1 0 0 0 0];	volScalarField; [0 0 0 0 0 0 0];	volScalarField; [0 0 0 0 0 0 0];	volScalarField; [1 -1 -2 0 0 0 0];
internal Field		uniform (0 0 0);	nonuniform List<scalar> 2991 ( 0 0...	uniform 0;	uniform 0;
Default	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
frontBack	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
wall	type wall;	type fixedValue; value uniform (0 0 0);	type zeroGradient;	type zeroGradient;	type buoyantPressure; value uniform 0;
open	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
defaultFac es	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;

wallのデータを全て選択し、  
右クリックして「コピー」

wallのpatchTypeを選択し、  
右クリックして「貼付け」  
(画面は既に貼付けた状態)

同様にしてデータを  
コピーして境界条件を  
作成し、保存する。

## 2-6. 境界条件の編集 (gridEditorで編集)

「コピ-」「貼付け」で作成した境界条件

	define patch (boundary)	U	alpha1	alpha1.org	p_rgh
field type		volVectorField;	volScalarField;	volScalarField;	volScalarField;
dimensions		[0 1 -1 0 0 0 0];	[0 0 0 0 0 0 0];	[0 0 0 0 0 0 0];	[1 -1 -2 0 0 0 0];
internal Field		uniform (0 0 0);	nonuniform List<scalar> 2991 ( 0 0...	uniform 0;	uniform 0;
Default	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;
frontBack	type patch;	type slip;	type zeroGradient;	type zeroGradient;	type zeroGradient;
wall	type wall;	type fixedValue; value uniform (0 0 0);	type zeroGradient;	type zeroGradient;	type buoyantPressure; value uniform 0;
open	type patch;	type pressureInletOutletVelocity; value uniform (0 0 0);	type inletOutlet; inletValue uniform 0; value uniform 0;	type inletOutlet; inletValue uniform 0; value uniform 0;	type totalPressure; p0 uniform 0; U U; phi phi; rho rho; psi none; gamma 1; value uniform 0;
defaultFaces	type patch;	type zeroGradient;	type zeroGradient;	type zeroGradient;	type zeroGradient;

```

boundary (~:/CAE/CAE-FO
ファイル(F) 編集(E) 表示(V) 検索(S)
開く 保存
boundary x boundary x
// *****
5
(
  Default
  {
    type          patch;
    nFaces        0;
    startFace     4963;
  }
  frontBack
  {
    type          patch;
    nFaces        1712;
    startFace     4963;
  }
  wall
  {
    type          patch;
    nFaces        248;
    startFace     6675;
  }
  open
  {
    type          patch;
    nFaces        78;
    startFace     6923;
  }
  defaultFaces
  {
    type          patch;
    nFaces        0;
    startFace     7001;
  }
)
// *****

```

nFaceが「0」のpatchは、黄色のハッキングで表示される。  
パッチ名を右クリックしてpatchを削除できるので、  
このパッチを削除する。

## 2-6. 境界条件の編集 (gridEditorで編集)

不要なpatchを削除した最終的なboundaryFieldの内容

	define patch (boundary)	U	alpha1	alpha1.org	p_rgh
field type		volVectorField;	volScalarField;	volScalarField;	volScalarField;
dimensions		[0 1 -1 0 0 0 0];	[0 0 0 0 0 0 0];	[0 0 0 0 0 0 0];	[1 -1 -2 0 0 0 0];
internal Field		uniform (0 0 0);	nonuniform List<scalar> 2991 ( 0 0...	uniform 0;	uniform 0;
frontBack	type patch;	type slip;	type zeroGradient;	type zeroGradient;	type zeroGradient;
wall	type wall;	type fixedValue; value uniform (0 0 0);	type zeroGradient;	type zeroGradient;	type buoyantPressure; value uniform 0;
open	type patch;	type pressureInletOutletVelocity; value uniform (0 0 0);	type inletOutlet; inletValue uniform 0; value uniform 0;	type inletOutlet; inletValue uniform 0; value uniform 0;	type totalPressure; p0 uniform 0; U U; phi phi; rho rho; psi none; gamma 1; value uniform 0;

モデルの表裏面は、tutorialでは、2次元の解析の為、「empty」になっているが、今回のモデルでは、厚さ方向のメッシュ数が複数あるので、3次元解析になる。この為、この面を上記設定に変更した。

The image shows the TreeFoam GUI and a terminal window. The GUI window is titled "TreeFoam\_1.21-120122 (0)" and has a menu bar with "ファイル", "編集", "計算", "ツール", and "ヘルプ". The toolbar contains various icons, with two specific icons circled: a play button (labeled "1. クリック(計算開始)") and a multi-colored square icon (labeled "2. クリック(paraFoam起動)"). The main area of the GUI displays the OpenFoam environment path, the case directory, and the current case name "damBreak". Below this, there are buttons for "open" and "interFoam".

The terminal window is titled "caeuser@ubuntu1010V: ~/CAE/CAE-FOAM/test/damBreak-newModel" and shows the output of the simulation. The output includes Courant Number statistics, MULES solving for alpha1, and DICPCG solving for p\_rgh. The simulation ends with the message "End".

```
caeuser@ubuntu1010V: ~/CAE/CAE-FOAM/test/damBreak-newModel
Courant Number mean: 0.0499112 max: 0.355761
Interface Courant Number mean: 0.0125355 max: 0.320731
deltaT = 0.00230453
Time = 1

MULES: Solving for alpha1
Liquid phase volume fraction = 0.133477 Min(alpha1) = -7.99544e-21 Max(alpha1) = 0.999977
MULES: Solving for alpha1
Liquid phase volume fraction = 0.133477 Min(alpha1) = -2.98656e-21 Max(alpha1) = 0.999977
DICPCG: Solving for p_rgh, Initial residual = 0.0398051, Final residual = 0.00197855, No Iterations 2
time step continuity errors : sum local = 0.000838307, global = 1.1764e-06, cumulative = 0.000238244
DICPCG: Solving for p_rgh, Initial residual = 0.0149966, Final residual = 0.000634888, No Iterations 3
time step continuity errors : sum local = 0.000267043, global = 1.90338e-06, cumulative = 0.000240148
DICPCG: Solving for p_rgh, Initial residual = 0.00412557, Final residual = 9.64788e-08, No Iterations 81
time step continuity errors : sum local = 4.04715e-08, global = 7.0092e-10, cumulative = 0.000240149
ExecutionTime = 173.22 s  ClockTime = 380 s

End

caeuser@ubuntu1010V:~/CAE/CAE-FOAM/test/damBreak-newModel$
```

Below the terminal window, a 3D visualization of the dam break simulation is shown. The visualization displays a cross-section of a dam structure with a turbulent flow field. The color scale ranges from blue (low velocity) to red (high velocity), showing the turbulent wake behind the dam. A coordinate system with x, y, and z axes is visible in the bottom left corner of the visualization.

### 3. まとめ

今まで、SalomeMecaやOpenFOAMを使ってきた中で、「これがあったら便利」と言うものを作ってきた。

これが、TreeFoam、gridEditor。11/4/15から使用開始し、今は、手放せない状態。

1. SalomeMeca側でモデルを確認しながら定義したfaceやvolumeのグループをそのグループ名ごとメッシュ変換されるので、判り易い。  
(この方法で、220万要素まで変換できた。これ以上は未確認)
2. TreeFoamを使うことで、1アクションでメニューが開く。  
表示がTree構造なので、カレントディレクトリの場所のイメージがわく。
3. gridEditorを使うことで、fieldやpatchが多くある場合には確認が楽。  
高速検索しているので、geditよりも高速。  
internalFieldやboundaryFieldの編集もできるので、非常に便利。