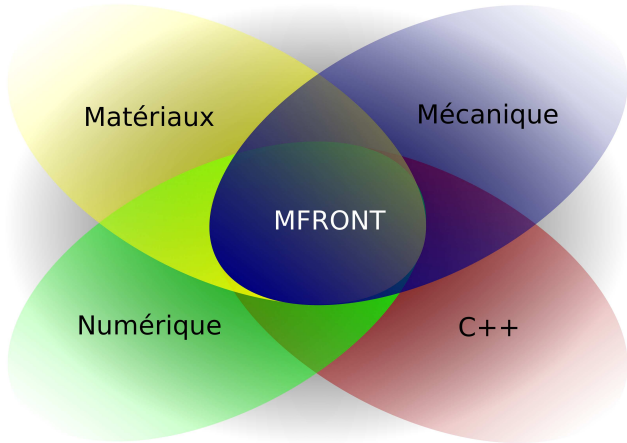
DE LA RECHERCHE À L'INDUSTRIE

www.cea.fr

Présentation de `mfront`



— T. Helfer

Sommaire

mfront **dans** pleiades

- **permettre** l'écriture de connaissances matériau :
  - les propriétés matériau (module d'Young, etc...) ;
  - les lois de comportement mécanique (viscoplasticité, plasticité, endommagement) ;
  - les modèles (gonflement, évolution physico-chimique) ;

`mfront` **dans** `pleiades`

- **permettre** l'écriture de connaissances matériau :
  - les propriétés matériau (module d'Young, etc...) ;
  - les lois de comportement mécanique (viscoplasticité, plasticité, endommagement) ;
  - les modèles (gonflement, évolution physico-chimique) ;
- **mutualiser** ces connaissances matériau :
  - entre les différentes études des applications de la plate-forme `pleiades` :
    - toutes les lois de comportement de la plate-forme vont être ré-écrites en `mfront` dès cette année ;
    - la plate-forme `pleiades` a crée une base de données nommée `sirius` qui a été adaptée pour utiliser des fichiers `mfront` en interne (+ de 100 matériaux différents) ;
  - avec d'autres codes :
    - quelque soit leur langage (fortran,C,C++,VBA, etc..) ;

mfront **dans** pleiades

- **permettre** l'écriture de connaissances matériau :
  - les propriétés matériau (module d'YOUNG, etc...) ;
  - les lois de comportement mécanique (viscoplasticité, plasticité, endommagement) ;
  - les modèles (gonflement, évolution physico-chimique) ;
- **mutualiser** ces connaissances matériau :
  - entre les différentes études des applications de la plate-forme `pleiades` :
    - toutes les lois de comportement de la plate-forme vont être ré-écrites en `mfront` dès cette année ;
    - la plate-forme `pleiades` a crée une base de données nommée `sirius` qui a été adaptée pour utiliser des fichiers `mfront` en interne (+ de 100 matériaux différents) ;
  - avec d'autres codes :
    - quelque soit leur langage (fortran,C,C++,VBA, etc..) ;
- **simplifier** le travail des utilisateurs :
  - numérique ;
  - informatique ;
  - minimiser le risque d'erreur

Nombre de lignes

	Propriétés	Loi de comportement	Modèle
Fichier mfront	33	**92**	22
Fichier généré	130	**1641**	556

- le gain, pour l'utilisateur, peut être conséquent ;
- on « automatise » au maximum pour réduire le risque d'erreurs en ce qui concerne la partie purement informatique de l'implantation ;
- ces « détails » informatiques peuvent avoir de nombreuses conséquences (portabilité, performances, etc..) ;
- une partie du code est de la « glue » pour s'adapter au code/langage cible.

Des contextes logiciels variés

- pour les propriétés matériaux :
  - différents langages (C++,fortran,python) ;
- pour les lois de comportement mécanique :
  - différents solveurs (Cast3M,Aster,TMFFT, mtests,...) ;
- pour les modèles :
  - `germinal`, `licos` ;

Des contextes logiciels variés

- pour les propriétés matériaux :
  - différents langages (C++,fortran,python) ;
- pour les lois de comportement mécanique :
  - différents solveurs (Cast3M,Aster,TMFFT, mtests,...) ;
- pour les modèles :
  - `germinal`, `licos` ;

`mfront` propose la notion d'**interface**

Des contextes logiciels variés

- pour les propriétés matériaux :
  - différents langages (C++,fortran,python) ;
- pour les lois de comportement mécanique :
  - différents solveurs (Cast3M,Aster,TMFFT, mtests,...) ;
- pour les modèles :
  - `germinal`, `licos` ;

`mfront` propose la notion d'**interface**

Le code généré dépend de l'interface choisie ! (on veut être performant !)

Propriétés matériaux

Conductivité thermique du combustible *UPuC*

- conductivité thermique du combustible *UPuC* :

$$k\left(T, p, \tau\right)$$

  - T est la température ;
  - p est la porosité ;
  - τ est le taux de combustion ;

Conductivité thermique du combustible *UPuC*

- conductivité thermique du combustible *UPuC* :

$$k\left(T, p, \tau\right)$$

  - T est la température ;
  - p est la porosité ;
  - τ est le taux de combustion ;
- introduction en 3 étapes :
  - écriture d'une fonction `UPuC_ThermalConductivity` ;
  - création d'une librairie `libUPuCMaterialProperties.so` ;
  - appel depuis `Cast3M` ;

Exemple en mfront

```
@Parser   MaterialLaw;
@Law      ThermalConductivity;
@Material UPuC;
@Author   Thomas Helfer;

@Output k;  //< changing the name of output

@Input T,p,Bu;                        //< inputs of the law

@Function{
  if (T<=773.15){
    k = (8.14e-6*T-0.010096882)*T+19.65063040915;
  } else {
    k = (-1.88e-6*T+0.009737044)*T+10.2405949657;
  }
  k *= (1.-p)/(1.+2.*p);
  k *= 1.-(0.02*Bu);
} // end of function
```

Exemple en mfront

```
@Parser   MaterialLaw;
@Law      ThermalConductivity;
@Material UPuC;
@Author   Thomas Helfer;

@Output k;  //< changing the name of output

@Input T,p,Bu;                        //< inputs of the law
T.setGlossaryName("Temperature");     //< pleiades name
p.setGlossaryName("Porosity");        //< pleiades name
Bu.setGlossaryName("BurnUp");         //< pleiades name

@PhysicalBounds T in [0:*[;   //< temperature physical bounds
@Bounds T in [0:2573.15];     //< temperature bounds
@PhysicalBounds p in [0:1];   //< porosity physical bounds
@PhysicalBounds Bu in [0:*[; //< burn-up   physicalbounds

@Function{
  if (T<=773.15){
    k = (8.14e-6*T-0.010096882)*T+19.65063040915;
  } else {
    k = (-1.88e-6*T+0.009737044)*T+10.2405949657;
  }
  k *= (1.-p)/(1.+2.*p);
  k *= 1.-(0.02*Bu);
} // end of function
```

Exemple en mfront

```
@Parser   MaterialLaw;
@Law      ThermalConductivity;
@Material UPuC;
@Author   Thomas Helfer;

@Output k;  //< changing the name of output

@Input T,p,Bu;                        //< inputs of the law
T.setGlossaryName("Temperature");     //< pleiades name
p.setGlossaryName("Porosity");        //< pleiades name
Bu.setGlossaryName("BurnUp");         //< pleiades name

@PhysicalBounds T in [0:*[;   //< temperature physical bounds
@Bounds T in [0:2573.15];     //< temperature bounds
@PhysicalBounds p in [0:1];   //< porosity physical bounds
@PhysicalBounds Bu in [0:*[; //< burn-up   physicalbounds

@Function{
  if (T<=773.15){
    k = (8.14e-6*T-0.010096882)*T+19.65063040915;
  } else {
    k = (-1.88e-6*T+0.009737044)*T+10.2405949657;
  }
  k *= (1.-p)/(1.+2.*p);
  k *= 1.-(0.02*Bu);
} // end of function
```

```
mfront --obuild --interface=castem UPuC_ThermalConductivity.mfront
```

Commentaires

- un fichier clair (avis subjectif) ;

Commentaires

- un fichier clair (avis subjectif) ;
- interfaces disponibles :
  - castem (!) ;
  - Excel (Visual Basic) ;
  - C/C++/fortran ;
  - python ;
  - octave ;
  - gnuplot ;
  - etc...

Commentaires

- un fichier clair (avis subjectif) ;
- interfaces disponibles :
  - castem (!) ;
  - Excel (Visual Basic) ;
  - C/C++/fortran ;
  - python ;
  - octave ;
  - gnuplot ;
  - etc...
- gestion facilitée des bornes des propriétés matériau ;

Commentaires

- un fichier clair (avis subjectif) ;
- interfaces disponibles :
  - castem (!) ;
  - Excel (Visual Basic) ;
  - C/C++/fortran ;
  - python ;
  - octave ;
  - gnuplot ;
  - etc...
- gestion facilitée des bornes des propriétés matériau ;
- gestion facilitée des dépendances entre propriétés matériau ;

Commentaires

- un fichier clair (avis subjectif) ;
- interfaces disponibles :
  - castem (!) ;
  - Excel (Visual Basic) ;
  - C/C++/fortran ;
  - python ;
  - octave ;
  - gnuplot ;
  - etc...
- gestion facilitée des bornes des propriétés matériau ;
- gestion facilitée des dépendances entre propriétés matériau ;
- support de la procédure de compilation :

Commentaires

- un fichier clair (avis subjectif) ;
- interfaces disponibles :
  - castem (!) ;
  - Excel (Visual Basic) ;
  - C/C++/fortran ;
  - python ;
  - octave ;
  - gnuplot ;
  - etc...
- gestion facilitée des bornes des propriétés matériau ;
- gestion facilitée des dépendances entre propriétés matériau ;
- support de la procédure de compilation :
- interaction avec la base de données matériau `sirius` :
  - en entrée ;
  - en sortie ;

Utilisation dans Cast3M (version pleiades)

```
* Création d'un modèle thermique isotrope
ModT1 = 'MODELISER' s1 'THERMIQUE' 'ISOTROPE' ;

* Création d'une table contenant les données relatives
* à la propriété externe :
* - 'MODELE' contient le nom de la fonction appelée
* - 'LIBRAIRIE' contient le nom de la librairie externe
* dans laquelle cette fonction est définie
* - 'VARIABLES' contient la liste des paramètres dont dépend
* la fonction appelée
Tmat = 'TABLE';
Tmat. 'MODELE' = 'UPuC_ThermalConductivity' ;
Tmat. 'LIBRAIRIE' = 'libUPuCMaterialProperties.so' ;
Tmat. 'VARIABLES' = 'MOTS' 'T' 'PORO' 'FIMA';

* Création du matériau.
MatT1 = 'MATERIAU' ModT1 'K' Tmat;
```

Utilisation dans Cast3M (version pleiades)

```
* Création d'un modèle thermique isotrope
ModT1 = 'MODELISER' s1 'THERMIQUE' 'ISOTROPE' ;

* Création d'une table contenant les données relatives
* à la propriété externe :
* - 'MODELE' contient le nom de la fonction appelée
* - 'LIBRAIRIE' contient le nom de la librairie externe
* dans laquelle cette fonction est définie
* - 'VARIABLES' contient la liste des paramètres dont dépend
* la fonction appelée
Tmat = 'TABLE';
Tmat. 'MODELE' = 'UPuC_ThermalConductivity' ;
Tmat. 'LIBRAIRIE' = 'libUPuCMaterialProperties.so' ;
Tmat. 'VARIABLES' = 'MOTS' 'T' 'PORO' 'FIMA';

* Création du matériau.
MatT1 = 'MATERIAU' ModT1 'K' Tmat;
```

- utilisation transparente dans les procédures classiques (PASAPAS) ;

Utilisation dans Cast3M (version pleiades)

```
* Création d'un modèle thermique isotrope
ModT1 = 'MODELISER' s1 'THERMIQUE' 'ISOTROPE' ;

* Création d'une table contenant les données relatives
* à la propriété externe :
* - 'MODELE' contient le nom de la fonction appelée
* - 'LIBRAIRIE' contient le nom de la librairie externe
* dans laquelle cette fonction est définie
* - 'VARIABLES' contient la liste des paramètres dont dépend
* la fonction appelée
Tmat = 'TABLE';
Tmat. 'MODELE' = 'UPuC_ThermalConductivity' ;
Tmat. 'LIBRAIRIE' = 'libUPuCMaterialProperties.so' ;
Tmat. 'VARIABLES' = 'MOTS' 'T' 'PORO' 'FIMA';

* Création du matériau.
MatT1 = 'MATERIAU' ModT1 'K' Tmat;
```

- utilisation transparente dans les procédures classiques (PASAPAS) ;
- utilisation simple de lois multi-variables :
  - les paramètres doivent être définies par des « chargements » ;

Modèles

Modèles

```
@Parser   Model;
@Model    SolidSwellingModel;
@Material UPuC;
@Author   Helfer Thomas;
@Date     06 Déc. 2007;

@Output s;
s.setGlossaryName("SolidSwelling");
s.setDefaultInitialValue(0.);
s.setDepth(1);

@Input Bu;
Bu.setGlossaryName("BurnUp");
Bu.setDepth(1);

@Input p;
p.setGlossaryName("Porosity");
p.setDepth(1);

@Function compute
{
  const real coef1 = ...;
  const real coef2 = ...;
  const real p_    = 0.5*(p+p_1);
  s = s_1 + coef1*exp(coef2-p_)*(Bu-Bu_1);
} // end of function compute
```

- aujourd'hui surtout utilisé dans les applications `pleiades` ;

Annexes

Portabilité

- `tfel` est développé sous `LiNuX` :
  - pas de problème de portabilité connu ;
  - seule plate-forme « officielle ».
- `tfel` a été porté sous `Windows` (mingw) :
  - `mfront` fonctionne sans soucis ;
  - `mfront` ne gère le processus de compilation sauf dans l'environnement MSYS ;
  - ne fonctionne pas avec Visual Studio... pour l'instant !
- `tfel` a été porté sous différents `Unix` :
  - `freebsd`, `opensolaris` ;
  - pas de problèmes particuliers ;

Compilateurs

- `tfel` est développé et testé en utilisant les compilateurs suivants :
  - g++, GNU, libre, versions 3.4 à 4.7)
  - clang++, LLVM, libre, versions 3.0 et 3.2)
  - ekopath, PathScale, libre (ou pas)
  - icpc, Intel propriétaire