Guide For Extension:

Real-fluid thermophysicalModels: An OpenFOAM-based library for reacting flow simulations at high pressure

Danh Nam Nguyen, Ki Sung Jung, Jae Won Shim, Chun Sang Yoo

To cite this article: D. N. Nguyen, K. S. Jung, J. W. Shim, C. S. Yoo, Real-fluid thermophysicalModels: An OpenFOAM-based library for reacting flow simulations at high pressure, Computer Physics Communications (2021)(submitted).

Note: This document presents the implementation guide for extension of the real-fluid *thermophysicalModels* library based on the implemented real-fluid models in *1-Implementation Guide-SRKchungTaka.pdf* file. We take the Peng-Robinson equation of state [1] as an example from which readers can refer to implement a new real fluid model or a new set of real fluid thermophysical models using our proposed method for mixture model.

Contents

1. A procedure to extend real-fluid thermophysicalModels library

There is no general procedure for extension of the real-fluid *thermophysicalModels* library since it depends on the combination of the models that you want to implement with the existing models in the library. In this document, we illustrate the implementation of Peng-Robinson (PR) EoS combining with Chung and Takahashi models for transport properties. For this combination, several classes need to be created, as shown in Figure. 1. There are two major steps for this implementation.

- Step 1: Create runtime selectable packages including PR EoS model;
- Step 2: Implement PR EoS model;

New classes have been created:

```
// In thermophysicalModels/specie directory:
- specie/equationOfState/PengRobinson/PengRobinson
// In thermophysicalModels/reaction directory:
- reactionThermo/mixtures/PRchungTakaMixture
- reactionThermo/mixtures/PRchungTakaReactingMixture
// In thermophysicalModels/chemistryModel directory:
- chemistryModel/chemistryModel/PRchungTakaStandardChemistryModel
```

2. Create runtime selectable packages for PR EoS model

2.1. Making source files

Suppose the real fluid *thermophysicalModels* library including SRK EoS, Chung's model for thermal conductivity and dynamic viscosity with Takahashi's correction for binary diffusion coefficients at high pressure has been created in your directory e.g., *yourDirectory* with a path variable as:

```
LIB_REALFLUID_SRC=~/OpenFOAM/yourDirectory/src/
```

Go to */thermophysicalModels/specie/equationOfState/* directory.

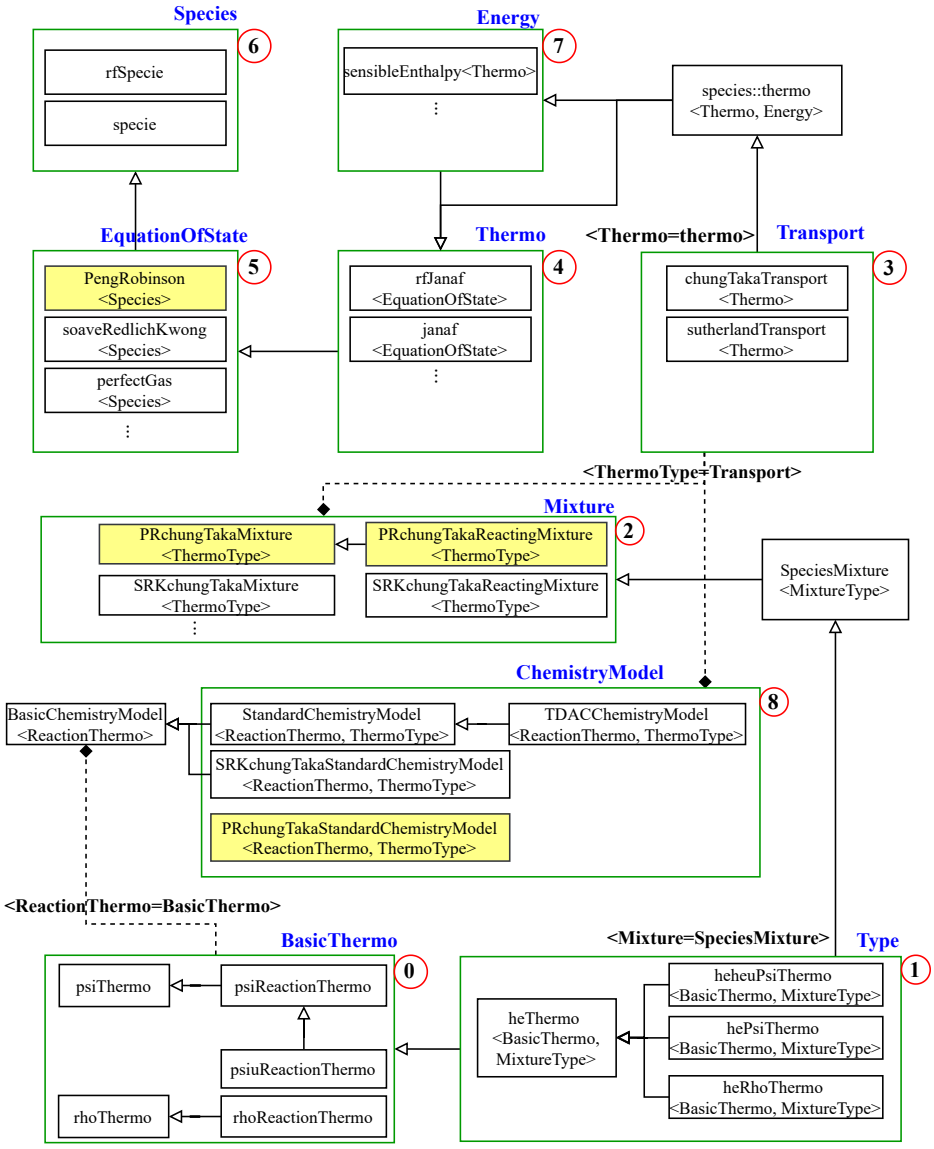

Figure 1: The class diagram of the real-fluid *thermophysicalModels* library in OpenFOAM-6. Yellow boxes are classes need to be created representing to new PR EoS model and its associated classes. The arrow-line denotes the inheritance relationship in which the direction of arrow is from a subclass to its base class. A dashed line denotes a class-class or class-solver interface in which one class is used in another or in a solver

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/specie/
   equationOfState/
```

Create *PengRobinson* class by copying from *soaveRedlichKwong* class.

```
cp -rf soaveRedlichKwong PengRobinson
cd PengRobinson
mv soaveRedlichKwong.H PengRobinson.H
mv soaveRedlichKwong.C PengRobinson.C
mv soaveRedlichKwongI.H PengRobinsonI.H
```

Open these files and replace *soaveRedlichKwong* by *PengRobinson*. To make sure you do not miss any thing you should use a command to automatically find and replace a string, for instance use the following command if you are using vim text editor.

```
vi PengRobinson.H
:%s/soaveRedlichKwong/PengRobinson/g
```

Do the same to create *PRchungTakaMixture*, *PRchungTakaReactingMixture*, and *PRchungTakaStandardChemistryModel* classes from *SRKchungTakaMixture*, *SRKchungTakaReactingMixture*, and *SRKchungTakaStandardChemistryModel* classes inside */reactionThermo/mixtures* and *chemistryModel/chemistryModel* directories, respectively. For instance:

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/reactionThermo/
   mixtures/
cp -rf SRKchungTakaMixture PRchungTakaMixture
cd PRchungTakaMixture
mv SRKchungTakaMixture.H PRchungTakaMixture.H
mv SRKchungTakaMixture.C PRchungTakaMixture.C
vi PRchungTakaMixture.H
:%s/SRKchungTakaMixture/PRchungTakaMixture/g
...
```

Note that change *SRKchungTakaMixture* into *PRchungTakaMixture* inside the *PRchungTakaReactingMixture* class since *PRchungTakaMixture* class is its base class. Then change *SRKchungTakaReactingMixture* into *PRchungTakaReactingMixture* inside the *PRchungTakaStandardChemistryModel* class, and the name at runtime of the *PRchungTakaStandard-*

ChemistryModel class should be specified correctly as follows:

```
// In PRchungTakaStandardChemistryModel.H file
    //- Runtime type information
    TypeName("PRchungTakaStandard");
```

2.2. Making runtime selectable thermophysical model package

Our goal is create four real fluid *thermo-packages* including PR EoS model in which the type of system can be either *hePsiThermo* (*psi*-based) or *heRhoThermo* (*rho*-based) and the energy type can be either *sensibleEnthalpy* or *sensibleInternalEnergy* as following:

```
thermoType
{
    type            hePsiThermo;        //or "heRhoThermo"
    mixture         PRchungTakaReactingMixture;
    transport       chungTaka;
    thermo          rfJanaf;
    energy          sensibleEnthalpy;   //or "sensibleInternalEnergy"
    equationOfState PengRobinson;
    specie          rfSpecie;
}
```

2.2.1. Making interface inside thermophysicalModels library

Modify these following files in order.

```
// =============== The list files ======================================
// In thermophysicalModels/specie directory
1. thermophysicalModels/specie/include/thermoPhysicsTypes.H
2. thermophysicalModels/specie/include/reactionTypes.H
3. thermophysicalModels/specie/reaction/reactions/makeReaction.H
4. thermophysicalModels/specie/reaction/reactions/makeReactions.C
//--> compile specie to make specie.so

// In thermophysicalModels/basic directory
//--> compile basic to make fluidThermophysicalModels.so
```

```
// In thermophysicalModels/reactionThermo directory
1. thermophysicalModels/reactionThermo/psiReactionThermo/
   psiReactionThermos.C
2. thermophysicalModels/reactionThermo/rhoReactionThermo/
   rhoReactionThermos.C
3. thermophysicalModels/reactionThermo/chemistryReaders/chemistryReader/
   makeChemistryReaders.C
//--> compile reactionThermo to make reactionThermophysicalModels.so

// In thermophysicalModels/chemistryModel directory
1. thermophysicalModels/chemistryModel/chemistryModel/BasicChemistryModel/
   BasicChemistryModels.C
2. thermophysicalModels/chemistryModel/chemistryModel/basicChemistryModel/
   basicChemistryModelTemplates.C
  //new created files
3. thermophysicalModels/chemistryModel/chemistrySolver/chemistrySolver/
   makeRealFluidChemistrySolverTypes.H
4. thermophysicalModels/chemistryModel/chemistrySolver/chemistrySolver/
   makeRealFluidChemistrySolvers.C
5. thermophysicalModels/chemistryModel/Make/files (+ options)
//--> compile chemistryModel to make chemistryModel.so
// ========================================================================
```

a. Modification in */yourDirectory/src/thermophysicalModels/specie* directory:

In *include/thermoPhysicsTypes.H* file, include the header files of new classes.

```
...
#include "PengRobinson.H"
```

Then make the shorthand names of combinations for set of real-fluid models.

```
typedef
chungTakaTransport
<
    species::thermo
```

```
    <
        rfJanafThermo
        <
            PengRobinson<rfSpecie>
        >,
        sensibleEnthalpy
    >
>
chungTakaRealJprHThermoPhysics;
```

and

```
typedef
chungTakaTransport
<
    species::thermo
    <
        rfJanafThermo
        <
            PengRobinson<rfSpecie>
        >,
        sensibleInternalEnergy
    >
>
chungTakaRealJprEThermoPhysics;
```

Note that the these two shorthand names *chungTakaRealJprHThermoPhysics* and *chungTakaRealJprEThermoPhysics* will be used many times later on. We will refer these two combinations are two THERMPHYS types.

In *include/reactionTypes.H* file, create new alias of reaction types based on two THERMPHYS types defined in *thermophysicalTypes.H* file as follows.

```
...
typedef Reaction<chungTakaRealJprHThermoPhysics> chungTakaRealJprHReaction
    ;
...
```

```
typedef Reaction<chungTakaRealJprEThermoPhysics> chungTakaRealJprEReaction
    ;
...
```

In *reaction/reactions/makeReaction.H* file, include the header files of new classes.

```
...
#include "PengRobinson.H"
```

In *reaction/reactions/makeReactions.C* file, create new *makeReactions* macros as follows.

```
...
makeReactions(chungTakaRealJprHThermoPhysics, chungTakaRealJprHReaction)
...
makeReactions(chungTakaRealJprEThermoPhysics, chungTakaRealJprEReaction)
...
```

Compile to make *specie.so*.

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/specie
wclean
wmake libso
```

If errors occur, check carefully your steps again to make sure you do not miss any thing.

b. Modification in */yourDirectory/src/thermophysicalModels/basic* directory:

Compile to make *fluidThermophysicalModels.so*.

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/basic
wclean
wmake libso
```

If errors occur, check carefully your steps again to make sure you do not miss any thing.

c. Modification in *yourDirectory/src/thermophysicalModels/reactionThermo* directory:

Go to */yourDirectory/src/thermophysicalModels/reactionThermo/* directory.

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/reactionThermo/
```

In *psiReactionThermo/psiReactionThermos.C*, include the header files of new classes and then create new *makeThermoPhysicsReactionThermos* macros as follows:

```
...
#include "PengRobinson.H"
#include "PRchungTakaMixture.H"
#include "PRchungTakaReactingMixture.H"
...
// real fluid mixture thermo for sensible enthalpy
makeThermoPhysicsReactionThermos
(
    psiThermo,
    psiReactionThermo,
    hePsiThermo,
    PRchungTakaMixture,
    chungTakaRealJprHThermoPhysics
);

// real fluid mixture thermo for internal energy
makeThermoPhysicsReactionThermos
(
    psiThermo,
    psiReactionThermo,
    hePsiThermo,
    PRchungTakaMixture,
    chungTakaRealJprEThermoPhysics
);

// real fluid mixture reaction thermo for sensible enthalpy
makeThermoPhysicsReactionThermos
(
    psiThermo,
    psiReactionThermo,
    hePsiThermo,
    PRchungTakaReactingMixture,
    chungTakaRealJprHThermoPhysics
);
```

```
// real fluid mixture reaction thermo for internal energy
makeThermoPhysicsReactionThermos
(
    psiThermo,
    psiReactionThermo,
    hePsiThermo,
    PRchungTakaReactingMixture,
    chungTakaRealJprEThermoPhysics
);
...
```

In *rhoReactionThermo/rhoReactionThermos.C*, include the header files of new classes and then create new *makeThermoPhysicsReactionThermos* macros as follows:

```
...
#include "PengRobinson.H"
#include "PRchungTakaMixture.H"
#include "PRchungTakaReactingMixture.H"
...
// real fluid mixture thermo for internal energy
makeThermoPhysicsReactionThermos
(
    rhoThermo,
    rhoReactionThermo,
    heRhoThermo,
    PRchungTakaMixture,
    chungTakaRealJprEThermoPhysics
);

// real fluid mixture thermo for sensible enthalpy
makeThermoPhysicsReactionThermos
(
    rhoThermo,
    rhoReactionThermo,
```

```
    heRhoThermo,
    PRchungTakaMixture,
    chungTakaRealJprHThermoPhysics
);

// real fluid mixture reaction thermo for internal energy
makeThermoPhysicsReactionThermos
(
    rhoThermo,
    rhoReactionThermo,
    heRhoThermo,
    PRchungTakaReactingMixture,
    chungTakaRealJprEThermoPhysics
);

// real fluid mixture reaction thermo for sensible enthalpy
makeThermoPhysicsReactionThermos
(
    rhoThermo,
    rhoReactionThermo,
    heRhoThermo,
    PRchungTakaReactingMixture,
    chungTakaRealJprHThermoPhysics
);
...
```

In *chemistryReaders/chemistryReader/makeChemistryReaders.C* file, create new *makeChemistryReader* and *makeChemistryReaderType* macros as follows:

```
...
makeChemistryReader(chungTakaRealJprHThermoPhysics);
makeChemistryReaderType(foamChemistryReader,
   chungTakaRealJprHThermoPhysics);
...
makeChemistryReader(chungTakaRealJprEThermoPhysics);
```

```
makeChemistryReaderType(foamChemistryReader,
    chungTakaRealJprEThermoPhysics);
...
```

Compile to make *reactionThermophysicalModels.so*.

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/reactionThermo
wclean
wmake libso
```

If errors occur, check carefully your steps again to make sure you do not miss any thing.

d. Modification in */yourDirectory/src/thermophysicalModels/chemistryModel* directory:

Goto *thermophysicalModels/chemistryModel/chemistryModel* directory.

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/chemistryModel/
    chemistryModel/
```

In *chemistryModel/chemistryModel/BasicChemistryModel/BasicChemistryModels.C* file, include the header files of new classes and then create new *makeChemistryModelType* macros as follows:

```
#include "PRchungTakaStandardChemistryModel.H"
...
makeChemistryModelType
(
    PRchungTakaStandardChemistryModel,
    psiReactionThermo,
    chungTakaRealJprHThermoPhysics
);

makeChemistryModelType
(
    PRchungTakaStandardChemistryModel,
    rhoReactionThermo,
    chungTakaRealJprHThermoPhysics
);
```

```
makeChemistryModelType
(
    PRchungTakaStandardChemistryModel,
    psiReactionThermo,
    chungTakaRealJprEThermoPhysics
);

makeChemistryModelType
(
    PRchungTakaStandardChemistryModel,
    rhoReactionThermo,
    chungTakaRealJprEThermoPhysics
);
...
```

In *chemistryModel/chemistryModel/basicChemistryModel/basicChemistryModelTemplates.C* file, change the *methodName* variable to be abe to recognize the name of new *PRchungTaka-StandardChemistryModel* class created so far as follows:

```
...
 76   const word& methodName
 77   (
 78     chemistryTypeDict.lookupOrDefault<word>
 79     (
 80         "method",
 81         chemistryTypeDict.lookupOrDefault<bool>("TDAC", false)
 82         ? "TDAC"
 83         : chemistryTypeDict.lookupOrDefault<bool>("SRKchungTakaStandard"
    , false)
 84         ? "SRKchungTakaStandard"
 85         : chemistryTypeDict.lookupOrDefault<bool>("PRchungTakaStandard",
     false)
 86         ? "PRchungTakaStandard"
 87         : "standard"
 88     )
```

```
89  );
...
```

Goto *thermophysicalModels/chemistryModel/chemistrySolver* directory.

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/chemistryModel/
   chemistrySolver/chemistrySolver/
```

In *makeRealFluidChemistrySolverTypes.H* file, add the following code to define new macros for building chemistry solvers for PR EoS:

```
//*---------------------------------------------------------------------*//
...
#include "PRchungTakaStandardChemistryModel.H"
...
// * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * * //
#define makePRchungTakaChemistrySolverType(SS, Comp, Thermo)       \
                                                                   \
  typedef SS<PRchungTakaStandardChemistryModel<Comp, Thermo>>      \
  SS##Comp##Thermo;                                                \
                                                                   \
  defineTemplateTypeNameAndDebugWithName                           \
  (                                                                \
    SS##Comp##Thermo,                                              \
    (#SS"<" + word(PRchungTakaStandardChemistryModel<Comp,         \
    Thermo>::typeName_()) + "<" + word(Comp::typeName_())          \
    + "," + Thermo::typeName() + ">>").c_str(),                    \
    0                                                              \
  );                                                               \
                                                                   \
  BasicChemistryModel<Comp>::                                      \
    add##thermo##ConstructorToTable<SS##Comp##Thermo>              \
    add##SS##Comp##Thermo##thermo##ConstructorTo                   \
    ##BasicChemistryModel##Comp##Table_;                           \
                                                                   \
#define makePRchungTakaChemistrySolverTypes(Comp, Thermo)          \
                                                                   \
```

```
  makePRchungTakaChemistrySolverType                                    \
  (                                                                     \
    noChemistrySolver,                                                  \
    Comp,                                                               \
    Thermo                                                              \
  );                                                                    \
                                                                        \
  makePRchungTakaChemistrySolverType                                    \
  (                                                                     \
    EulerImplicit,                                                      \
    Comp,                                                               \
    Thermo                                                              \
  );                                                                    \
  makePRchungTakaChemistrySolverType                                    \
  (                                                                     \
    ode,                                                                \
    Comp,                                                               \
    Thermo                                                              \
  );
...
```

In *makeRealFluidChemistrySolvers.C* file, add the new macros as follows:

```
namespace Foam
{
  // Chemistry solvers based on sensibleEnthalpy
  ...
  makePRchungTakaChemistrySolverTypes(psiReactionThermo,
      chungTakaRealJprHThermoPhysics);
  makePRchungTakaChemistrySolverTypes(rhoReactionThermo,
      chungTakaRealJprHThermoPhysics);

  // Chemistry solvers based on sensibleInternalEnergy
  ...
  makePRchungTakaChemistrySolverTypes(psiReactionThermo,
      chungTakaRealJprEThermoPhysics);
```

```
  makePRchungTakaChemistrySolverTypes(rhoReactionThermo,
      chungTakaRealJprEThermoPhysics);
}
```

Compile to make *chemistryModel.so*.

```
cd ~/OpenFOAM/yourDirectory/src/thermophysicalModels/chemistryModel
wclean
wmake libso
```

If errors occur, check carefully your steps again to make sure you do not miss any thing.

2.2.2. Making interface outside of the thermophysicalModels library

The *TurbulenceModels* and *combustionModels* libraries must be updated since they utilize the *thermophysicalModels* library.

Go to */yourDirectory/src/TurbulenceModels/turbulenceModels/* directory and compile to make *turbulenceModels.so*.

```
cd ~/OpenFOAM/yourDirectory/src/TurbulenceModels/turbulenceModels
wclean
wmake libso
```

Go to */yourDirectory/src/TurbulenceModels/compressible/* directory and compile to make *compressibleTurbulenceModels.so*.

```
cd ~/OpenFOAM/yourDirectory/src/TurbulenceModels/compressible
wclean
wmake libso
```

Go to */yourDirectory/src/combustionModels/* directory and compile to make *combustionModels.so*.

```
cd ~/OpenFOAM/yourDirectory/src/combustionModels
wclean
wmake libso
```

2.2.3. Test new created runtime thermo-packages

Using the *realFluidReactingFoam* solver with provided test cases in *tutorials* directory to test the availability of created real-fluid runtime *thermo-packages* associated with PR EoS model.

Go to */yourDirectory/applications/solvers/realFluidReactingFoam/* directory and recompile it.

```
cd ~/OpenFOAM/yourDirectory/applications/solvers/realFluidReactingFoam/
wclean
wmake
```

Now, *realFluidReactingFoam* is ready to use.

Specify the *thermoType* dictionary in *constant/thermophysicalProperties* dictionary file to use new created runtime *thermo-packages* as follows:

```
thermoType  //This is a new thermo packages we have made so far
{
    type            hePsiThermo;
    mixture         PRchungTakaReactingMixture;
    transport       chungTaka;
    thermo          rfJanaf;
    energy          sensibleEnthalpy;
    equationOfState PengRobinson;
    specie          rfSpecie;
}
```

Specify the *chemistryType* dictionary in *constant/chemistryProperties* dictionary file to use new created runtime *thermo-packages* as follows:

```
chemistryType
{
    solver  EulerImplicit;
    method  PRchungTakaStandard; //new chemistry model for PR EoS
}
```

Then, execute the *realFluidReactingFoam*.

```
blockMesh
realFluidReactingFoam
```

It should be executed without error. So far, we have successfully created new runtime selectable *thermo-packages* for real-fluid THERMPHYS models associated with PR EoS.

Note that the equations in the source code of the *PengRobinson*, and *PRchungTakaMixture* classes of the new *thermo-packages* here is not the real PR EoS model since we have not implemented the actual real calculation of PR EoS yet.

3. Detail implementation of new real-fluid models

Since the implemented source code of real-fluid models are too long to be described in details in this document, reader are recommended referring directly to our source code for more convenience. The following classes should be replaced by our source files: *PengRobinson*, *PRchungTakaMixture*, and *PRchungTakaReactingMixture*.

4. Extend the real-fluid library for other models

Reader can apply the same technique as described above to implement any model into the real-fluid *thermophysicalModels* library. In the present work, we have implemented several models to provide more options for users. Please refer to our paper for these available options.

References

[1] D. Peng, D. Robinson, New two-constant equation of state, Ind. Eng. Chem. Fundam. 15 (1976) 59–64.