8.2 Extensions for solids

Providing coverage of polymers and solids in thermophysical properties

Issue

- THRM 1.0 or 1.1 interface specification do not handle distributed properties:
  - i.e. properties that may depend on other properties
  - Properties are only function of temperature, pressure and composition

- Solids, polymers are exhibiting properties such as:
  - Density depending on size

- Properties are often classified
  - Size intervals

Main features

- Matter is distributed in a number of classes
- Unit operations may change distribution of matter in classes: eg. grinding.
- The redistribution of fractions into existing classes can be achieved by applying a movement matrix.
- Each entry of the movement matrix represents a fraction of the material that is moved from one class to another of a distributed property.

Distributed properties

attribute name: **Particle Size**
dimension: **cm**
class defintion:

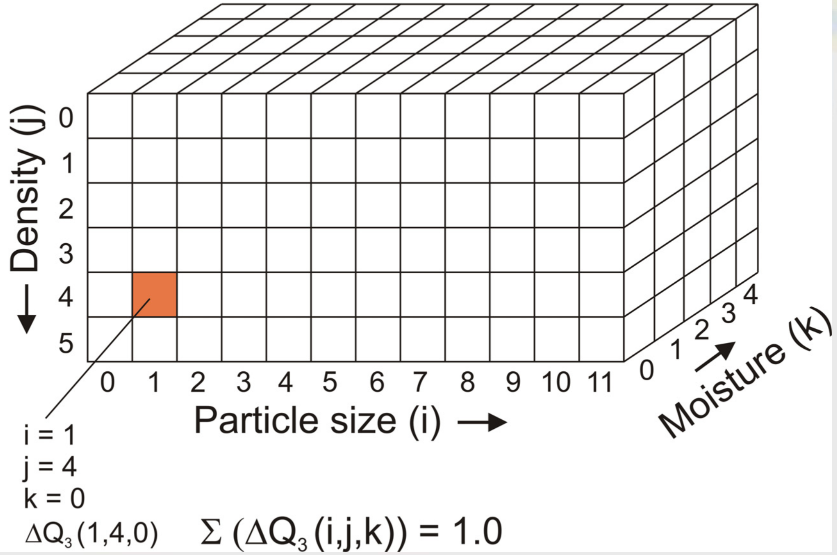
number	lower- /	upper bound
0	0,0000	0,0001
1	0,0001	0,0002
2	0,0002	0,0004
3	0,0004	0,0008
4	0,0008	0,0016
5	0,0016	0,0032
6	0,0032	0,0064
7	0,0064	0,0128
8	0,0128	0,0256
9	0,0256	0,0512
10	0,0512	0,1024
11	0,1024	0,2048

attribute name: **Density**
dimension: **kg/m³**
class defintion:

number	lower- /	upper bound
0	1000	1200
1	1200	1400
2	1400	1600
3	1600	1800
4	2000	2200
5	2400	2600

attribute name: **Moisture**
dimension: **kg/kg (dry)**
class defintion:

number	lower- /	upper bound
0	0.000	1.000
1	1.000	2.500
2	2.500	5.000
3	5.000	7.500
4	7.500	10.000



For each property the name, dimension (referring to units of measurement) and class interval definition has to be stored.

Non-distributed properties are defined by their name and dimension.

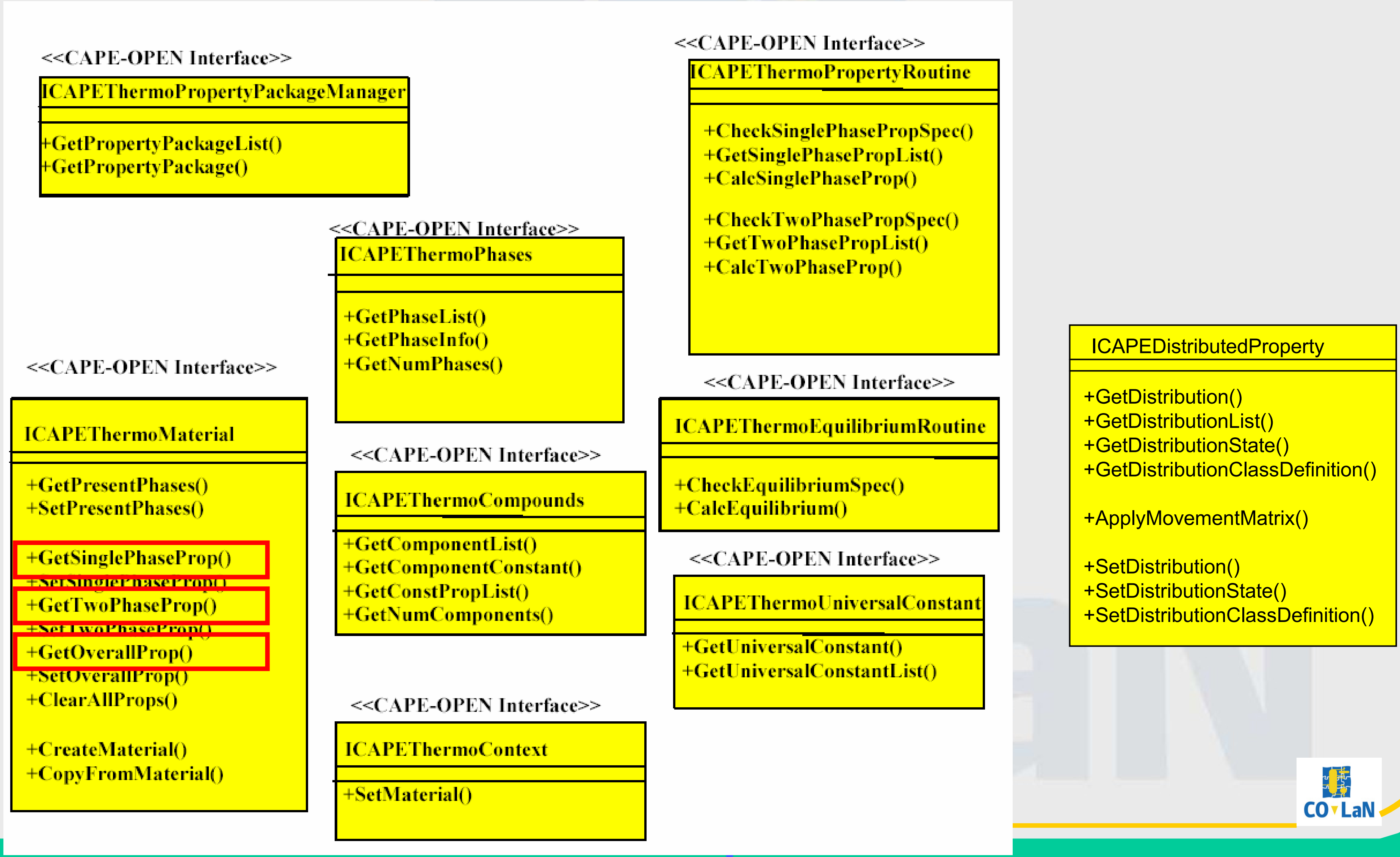
Movement matrix

to from	1	2	$\dots$	n
1	$k_{1,1}$	$\dots$	$\dots$	$k_{1,n}$
2	$k_{2,1}$	$\dots$	$\dots$	$k_{2,n}$
$\dots$	$\dots$	$\dots$	$\dots$	$\dots$
n	$k_{n,1}$	$\dots$	$\dots$	$k_{n,n}$

Solution proposed

- Extension will be applied only to THRM 1.1
- Solution proposed for THRM 1.1
  - Extend list of existing thermo properties to add distributed properties convenient for solids and polymers
  - Methods GetOverallProp, GetSinglePhaseProp and GetTwoPhaseProp may return an interface pointer.

Interface diagram 1.1



<<CAPE-OPEN Interface>>
ICAPEThermoPropertyPackageManager

+GetPropertyPackageList()
+GetPropertyPackage()

<<CAPE-OPEN Interface>>
ICAPEThermoMaterial

+GetPresentPhases()
+SetPresentPhases()

+GetSinglePhaseProp()
+SetSinglePhaseProp()
+GetTwoPhaseProp()
+SetTwoPhaseProp()
+GetOverallProp()
+SetOverallProp()
+ClearAllProps()

+CreateMaterial()
+CopyFromMaterial()

<<CAPE-OPEN Interface>>
ICAPEThermoPhases

+GetPhaseList()
+GetPhaseInfo()
+GetNumPhases()

<<CAPE-OPEN Interface>>
ICAPEThermoCompounds

+GetComponentList()
+GetComponentConstant()
+GetConstPropList()
+GetNumComponents()

<<CAPE-OPEN Interface>>
ICAPEThermoContext

+SetMaterial()

<<CAPE-OPEN Interface>>
ICAPEThermoPropertyRoutine

+CheckSinglePhasePropSpec()
+GetSinglePhasePropList()
+CalcSinglePhaseProp()

+CheckTwoPhasePropSpec()
+GetTwoPhasePropList()
+CalcTwoPhaseProp()

<<CAPE-OPEN Interface>>
ICAPEThermoEquilibriumRoutine

+CheckEquilibriumSpec()
+CalcEquilibrium()

<<CAPE-OPEN Interface>>
ICAPEThermoUniversalConstant

+GetUniversalConstant()
+GetUniversalConstantList()

ICAPEDistributedProperty

+GetDistribution()
+GetDistributionList()
+GetDistributionState()
+GetDistributionClassDefinition()

+ApplyMovementMatrix()

+SetDistribution()
+SetDistributionState()
+SetDistributionClassDefinition()

GetSinglePhaseProp method

Interface Name	**ICapeThermoMaterial**
Method Name	GetSinglePhaseProp
Returns	CapeError

Description

Retrieves single-phase non-constant property values for a mixture.

Arguments

Name	Type	Description
[in] *property*	CapeString	The identifier of the property for which values are requested. This must be one of the single-phase properties or derivatives. The standard identifiers are listed in sections ⌈7.5.5 and ⌈7.6.
[in] *phaseLabel*	CapeString	Phase label of the phase for which the property is required. The phase label must be one of the identifiers returned by the GetPresentPhases method of this interface.
[in] *basis*	CapeString	Basis of the result. Valid settings are: “Mass” for properties per unit mass; “Mole” for molar proper-ties or “Total” for total (extensive) properties. Use UNDEFINED as a place holder for a property for which basis does not apply. See section ⌈7.5.5 for details.
[ACTUALLYout] *results*	CapeVariant	Results vector (CapeArrayDouble) containing property value(s) in SI units or CapeInterface (see notes).

Solid extension: IDL available

```
 * Get the distributed property fraction
 * \param propertyList
List of property identifiers. The returned matrix is of dimension n = number of entries in
'propertyList'. 1st dimension of matrix contains class interval fractions of 1st property in
'propertyList', 2nd dimension -> class intervals of 2nd property in 'propertyList' etc.
 * \param phaseLabels
List of phase labels for which the properties are requested. For single phase properties
phaseLabel has one entry, for two phase properties two entries etc.
 * \param basis
Basis of the fraction in the returned matrix. Based on 'basis' the matrix will contain mass or
mole fractions.
 * \param distrProp
Pointer to CapeArrayDouble in which the matrix containing mass/mole fractions (<span
style="font-family:Symbol">D</span>Q<sub>3</sub>) describing the distribution of matter
about the class interval combinations of the requested properties is returned.
 * \return
 *   CAPE-OPEN exceptions: ECapeUnknownHR, ECapeInvalidArgumentHR,
ECapeThrmPropertyNotAvailableHR, ECapeNoImplHR
 */
[id(1), helpstring("method GetDistribution")]
HRESULT GetDistribution([in] CapeArrayString propertyList,
                        [in] CapeArrayString phaseLabels,
                        [in] CapeString basis,
                        [out, retval] CapeArrayDouble* distrProp);
```

Conclusion & perspectives

- Interface specification drafted
- Available from THRM SIG leader and CTO
- Implemented in SolidSim
- Comments are welcome