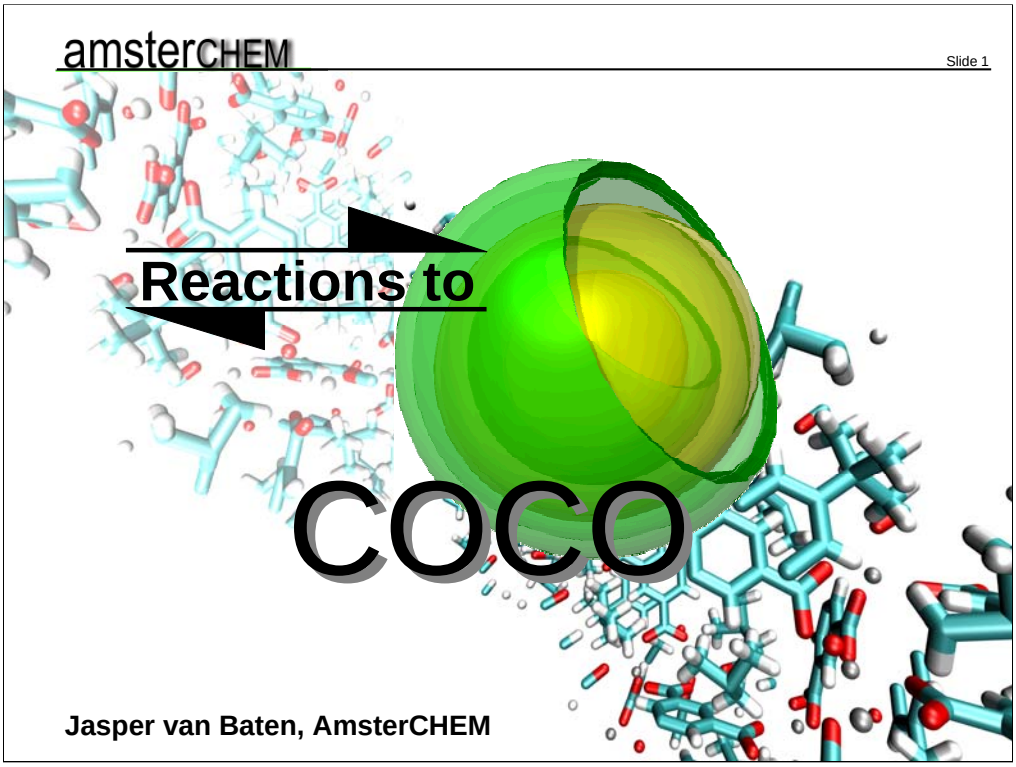


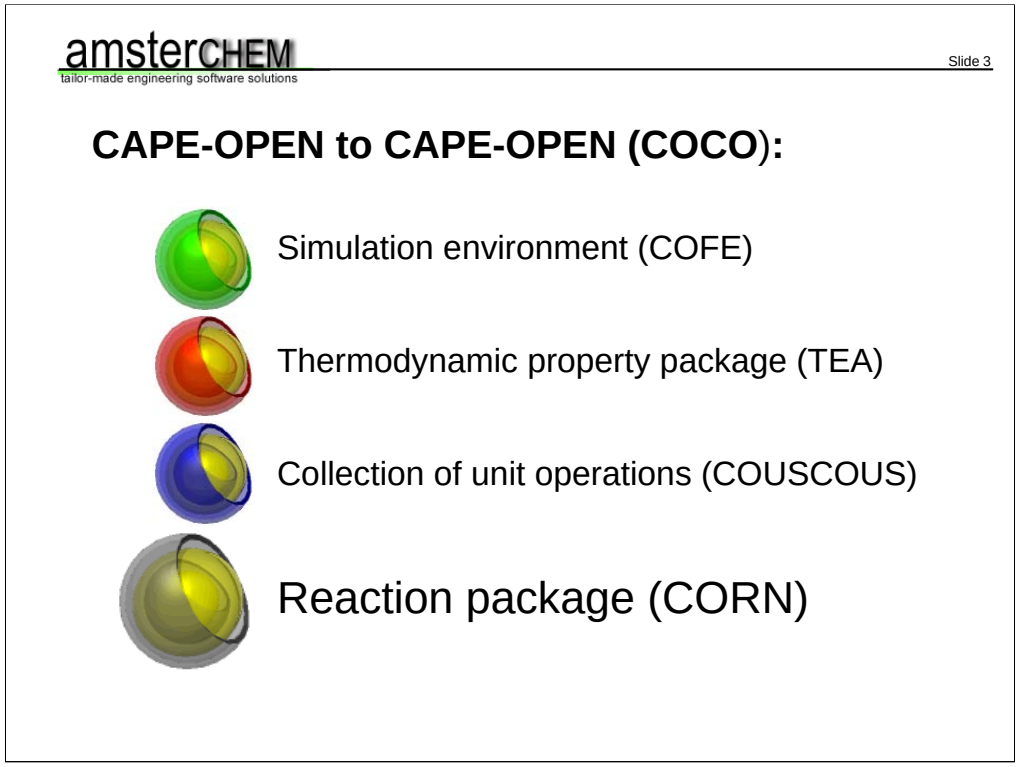
Good mornig. My name is Jasper van Baten. I represent AmsterCHEM. Today I will talk about Reaction Packages. I will do so using COCO.

amsterCHEM
tailor-made engineering software solutions

Presentation outline

- Reaction packages in the CAPE-OPEN framework
- Compound identification
- Heat of reaction
- Issues with standard specification
- Reactors supplied by COCO
- Support by COSEs is required

I will start off by showing how Reaction Packages fit into the CAPE-OPEN framework. Then I will highlight some challenges: compound identification and heat of reaction. There are other issues with the standard specification that I will shortly touch upon. I will show which reactors are implemented in COCO. And I will finish off by asking software vendors to implement support for the Reaction interfaces.



As most of you know, COCO is a collection of CAPE-OPEN software components. There is the CAPE-OPEN Flowsheeting Environment called COFE. The Thermodynamics for Engineering Applications module is called TEA. COUSCOUS provides COCO with unit operations. And what has been missing mostly from my previous presentations is what CORN exactly does. CORN stands for CAPE-OPEN Reaction Numerics, and implements Reaction Packages and a Reaction Package Manager. So what can we expect from a Reaction Package?

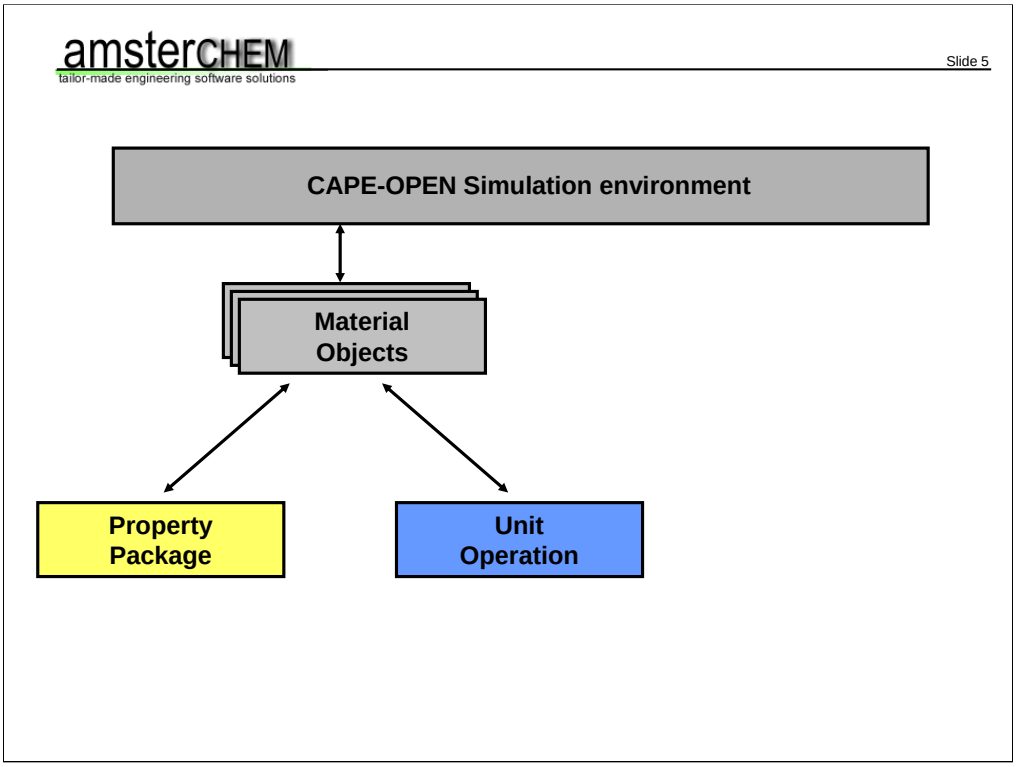
amsterCHEM
tailor-made engineering software solutions

Slide 4

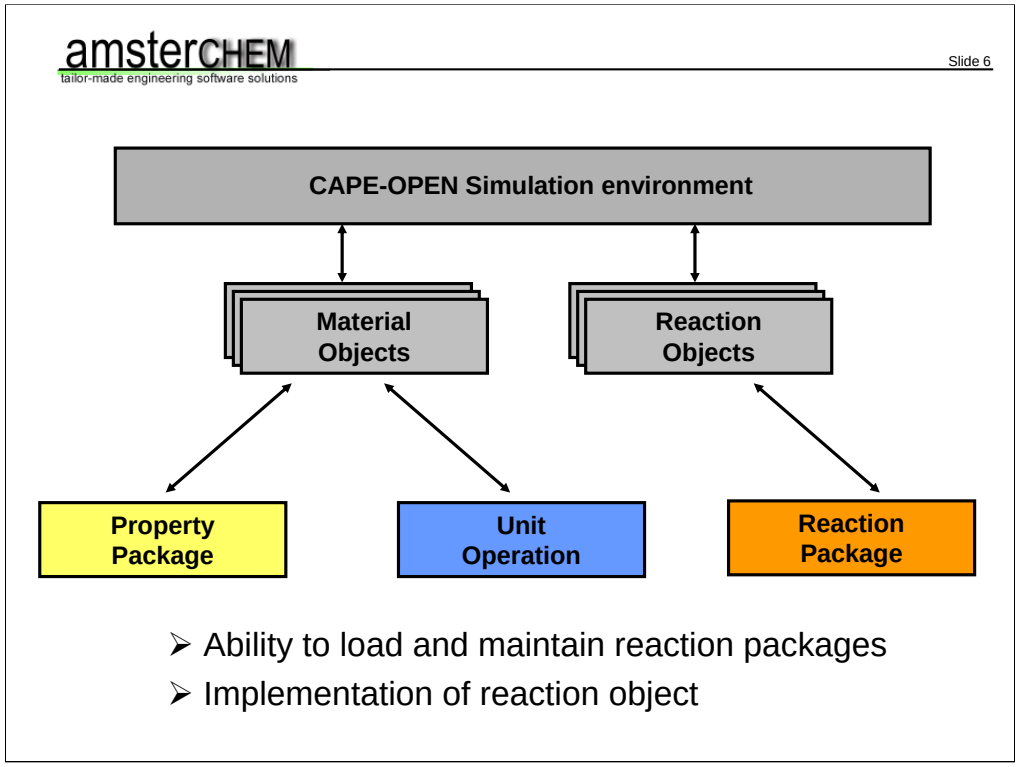
Reaction Packages contain:

- Reactions
- Reaction compounds
- Reaction stoichiometry
- Reaction type:
  - + kinetic or equilibrium
  - + homogeneous or heterogeneous
- Reaction phase
- Reaction rate / equilibrium constant
- Heat of reaction

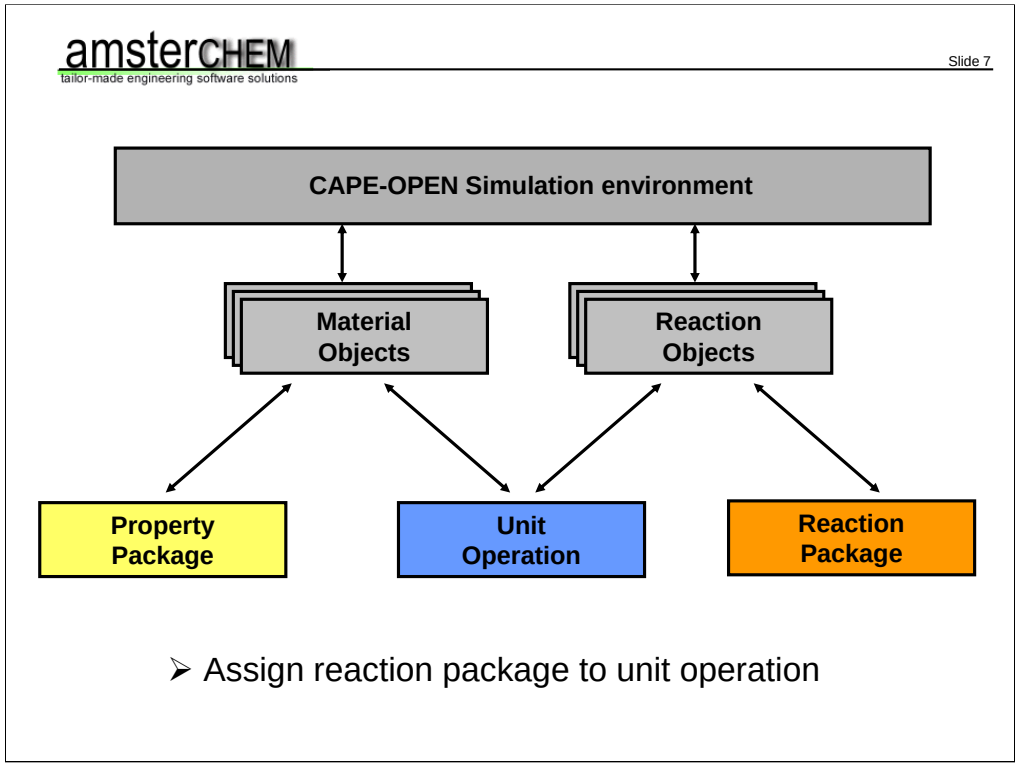
A Reaction Package will typically expose one or more reactions. For each reaction, we can obtain the compounds that play a role in it, the reaction stoichiometry, whether it is a kinetic or equilibrium reaction, whether it is a homogeneous or heterogeneous reaction - that is, whether catalyst is involved – and in which phase the reaction takes place. All of this defines the reaction chemistry. The Reaction Package will also supply us with some numbers that will allow us to do some calculations. For kinetic reactions, the reaction rate is available. For equilibrium reactions, the equilibrium constant can be calculated. For both types of reactions, the heat of reaction may or may not be available. More on heat of reaction later. Let’s first see how it all works.



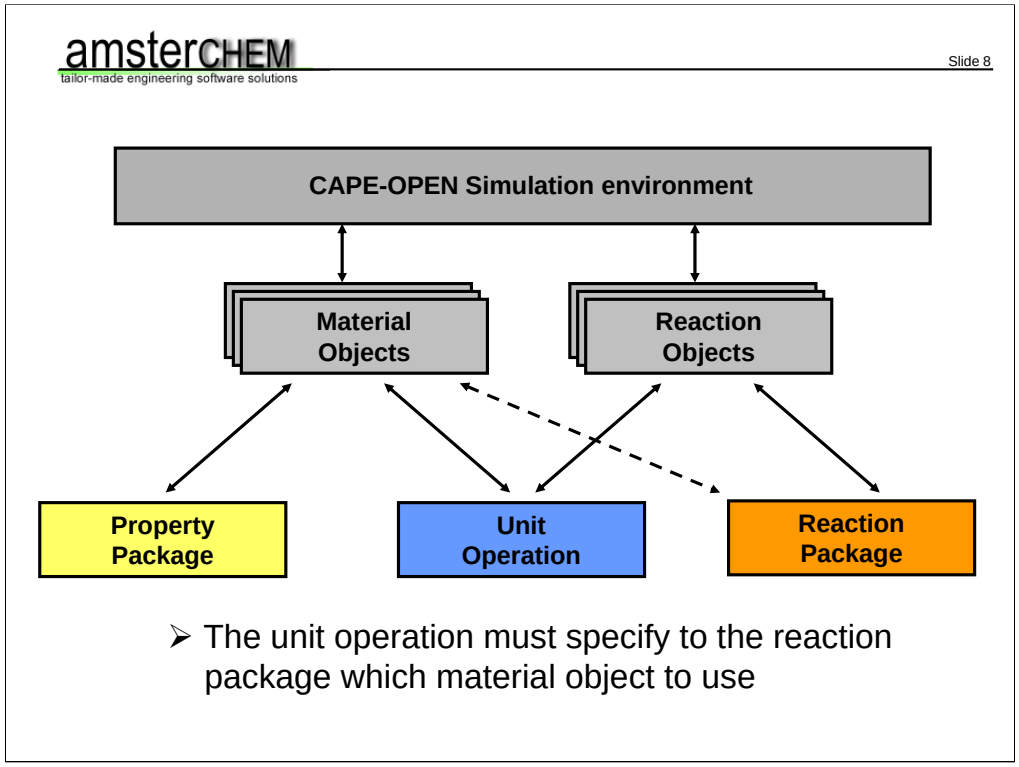
Most of us are familiar with this picture. We have a CAPE-OPEN simulation environment shown in the top. Material streams are represented by CAPE-OPEN Material Object interfaces. If an external CAPE-OPEN property package is loaded, it will talk to the simulation environment via the Material Object. Also any loaded CAPE-OPEN unit operations will talk to the simulation environment using the Material Objects. For example: the unit operation wants density for the vapor phase. It will ask the Material Object to do this calculation. The Material Object passes the calculation request on to the Property Package. The Property Package in turn will calculate density for the vapor phase. It will then call SetProp on the Material Object. The Material Object will store the value of the property. Control is returned to the Unit Operation. The Unit Operation can now access the value of vapor density by calling GetProp on the Material Object. It works more or less the same for reaction packages…



The Simulation Environment will implement Reaction Objects. The Reaction Package will talk to the Simulation Environment via the Reaction Object. So the Simulation Environment must implement Reaction Objects, and a way to load and maintain a collection of Reaction Packages. Loading a Reaction Package is exactly analogous to loading a Property Package: one can load a stand-alone Reaction Package, or one can use a Reaction Package Manager to create a Reaction Package.



The Unit Operation can access the Reaction Package via the Reaction Object. Any reaction property calculations will be passed on by the Reaction Object to the Reaction Package. The Reaction Package will set the calculated values back on the Reaction Object.
The Unit Operation already had access to the Material Object, since a Material Object was connected to one of its Ports. It did not have access to a Reaction Package yet. Therefore, a Unit Operation implements a Reaction Context, and the Simulation Environment must set the Reaction Package on the Unit Operation. Setting a Reaction Package on a Unit Operation is much like connecting a Material Object to a Unit Operation Port.



The Reaction Package must also talk to the Material Object: to calculate a reaction rate, the Reaction Package may require pressure, temperature, concentration or other physical properties. The Reaction Package therefore implements a Material Context. The Unit Operation must tell the Reaction Package which Material Object to use before doing any calculations.

Sounds simple, and it is. Let's look at some potential pitfalls…

amsterCHEM
tailor-made engineering software solutions

Slide 9

Potential pitfalls: Compound identification

- Property Package exposes: ID, CAS, MW, BP, formula, structure formula, IUPAC name, charge, …
- Reaction Package exposes ID, charge, CAS
- Reaction Package must match compounds
- Unit Operation must match compounds

Advice: Reaction package should adjust its IDs to match Material Object compound IDs

Let's start with compound identification. The Property Package is designed to expose a lot of information about a compound. According to te reactions interface specification, a compounds is defined by its ID, CAS number and charge. As not all compounds have a CAS number, and charge is obviously a bad choice for compound identification, we have only the ID to identify a compound. Keep in mind that the Reaction Package and the Property Package are two different objects, each with their own knowlegde about compounds. The Reaction Package will need to know which Material Object compound IDs matches its own compounds. Also the unit operation will need to know which Material Object compounds IDs match with the Reaction Package compounds IDs, for setting up the material balance. So it would be nice if the Reaction Package compound IDs and Material Object compound IDs would match. As the Material Object exposes a lot of compound information, and the Reaction Package must sort out a match between the Material Objects compounds and its own, it is in my view the best idea if the Reaction Package would adjust the Compounds IDs it exposes to match those of the Material Object, as soon as a Material Object is known. That is, at the SetMaterial call. This strategy is not clear from the interface specification.

Potential pitfalls: Heat of reaction

- Heat of reaction relates to heat of formation:

$$-\Delta H_r = \sum_i \upsilon_i \left(-H_f^0\right)_i$$

- "Enthalpy" may or may not include heat of formation
- "EnthalpyF" is not widely available (yet), but is sure to include heat of formation
- Balance with EnthalpyF requires flash: e.g. "PHF"

Another potential pitfall is heat of reaction. Should heat of reaction be included in the Unit Operation's heat balance or not? Good question. The heat of reaction is related to the heat of formation via the reaction stoichiometry as shown here. If we use property "Enthalpy" to set up the heat balance, it therefore depends on whether heat of formation is included in Enthalpy or not. We have no way of telling. If heat of formation is included in enthalpy, heat of reaction should not be taken into account, as it is taking into account implcitly. If Enthalpy however does not include heat of formation, heat of reaction is not accounted for and should be taken into account. To circumvent this problem, the thermo SIG came up last year with property "EnthalpyF". For this property, we can be sure that heat of formation is included. So no need anymore to take heat of reaction into account. Perfect. Thus, after setting up a heat balance with EnthalpyF, we end up with a value of EnthalpyF at the end of the reactor. Now we need to flash the outlets. In version 1.1 thermo, this is not a problem. We can just specify EnthalpyF as one of the flash constrains. For thermo version 1.0 however, we need a flash type that uses EnthalpyF rather than Enthalpy. So for example, PHF rather than PH.

All very nice. Unfortunately property EnthalpyF is not very widely supported at this point; to my knowlegde, TEA is the only one to implement it. So we are left with no other choice then asking the user which form of Enthalpy to use, and whether or not to include heat of reaction. Letting the user choose these kind of things is not always the best idea.

amsterCHEM
tailor-made engineering software solutions

Standard specification issues (I)

- Units of measure should be SI; kmol/h should be mol/s
- Undefined items should be removed or defined: GetBaseReactant, GetPhaseCompounds
- Derivatives are not available
- There is no Validate() method: validation at SetMaterial()
- Kinetic / Equilibrium reaction context: same
- Basis argument should be revised for some calls

There are more issues with the interface specification. COCO actually assumes that the interface specification will be adjusted, and does not necessarily stick to the specification. Most noticeably, COCO will assume SI units for the reaction rate. The interface specification currently says kmol/h rather than mol/s. I will not go over all items listed here now…

amsterCHEM
tailor-made engineering software solutions

Standard specification issues (II)

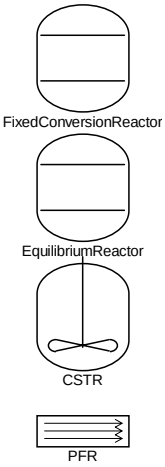
- GetReactionConcBasis:
  - + should apply to equilibrium – not kinetic – reactions
  - + additional identifiers: “molarity”, “concentration”, “moleFraction”, “massFraction”
- Reaction specification document is from 2003

but if you want to implement a plug or a socket for a reaction interface, please to not hesitate to discuss these matters with me. The Reactions Interface specification is from 2003 and is imply due for a maintenance service.

amsterCHEM
tailor-made engineering software solutions

Slide 13

Reactors supplied by COUSCOUS:



Fixed conversion reactor: specify conversion of reactions (parallel or series)

Equilibrium reactor: combined phase and reaction equilibrium calculation

Continuously stirred tank reactor: mix of equilibrium and kinetic reactions, well mixed

Plug flow reactor: kinetic reactions, one dimensional model

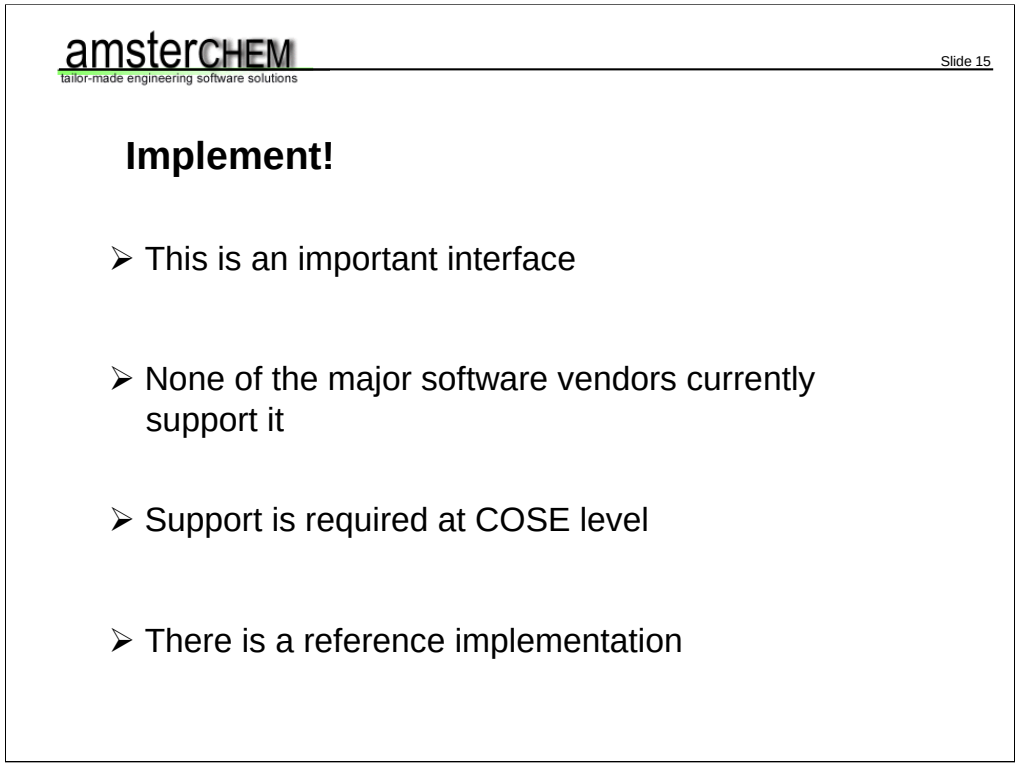
There is no point in having Reaction Packages without having Reactors. COUSCOUS will provide you with four reactors. For the Fixed Conversion Reactor, you can specify one or more reactions. Only the reaction chemistry is used, and you should specify for each reaction the conversion of one of the compounds. The Equilibrium Reactor will calculate a phase equilibrium combined with a reaction equilibrium. You can only use equilibrium reactions. If you want to use a mix of equilibrium reactions and kinetic reactions, you can use the Continuously Stirred Tank Reactor. The CSTR will assume a single well-mixed user-specified reaction phase.

The tubular reactor model is a one dimensional model. So all values in the radial direction are assumed equal. There is not support for back-mixing in the axial direction, so it is a Plug Flow Reactor. Only kinetic reactions can be specified.

Why use a Reaction Package:

- Uniform way of specification of reactions
- Reusable throughout the document in multiple reactors
- Reusable throughout multiple documents
- Reusable in various simulation environments (…)
- No duplication of data

To summarize: reaction packages provide you with a uniform way to specify reactions. You could argue that you can set up a Unit Operation that uses reactions without a Reaction Package, where the reactions are defined by the unit operation itself. However, when using a Reaction Package, you can re-use your reaction definitions for multiple unit operations. Or, within multiple documents. Or – the CAPE-OPEN paradigm – within multiple Simulation Environments. So why duplicate data? You should not. Use Reaction Packages instead.



Therefore: Yes, implement. This is an important interface, and currently there is very little support for it. None of the major software vendors to my knowledge provides support for it. It is especially important for the sockets to get implemented. Without, it is very hard to write CAPE-OPEN unit operations that perform reactions. If do you want to write a socket or a plug for a Reaction Package, you can of course test using COCO.

amsterCHEM
tailor-made engineering software solutions

Slide 16

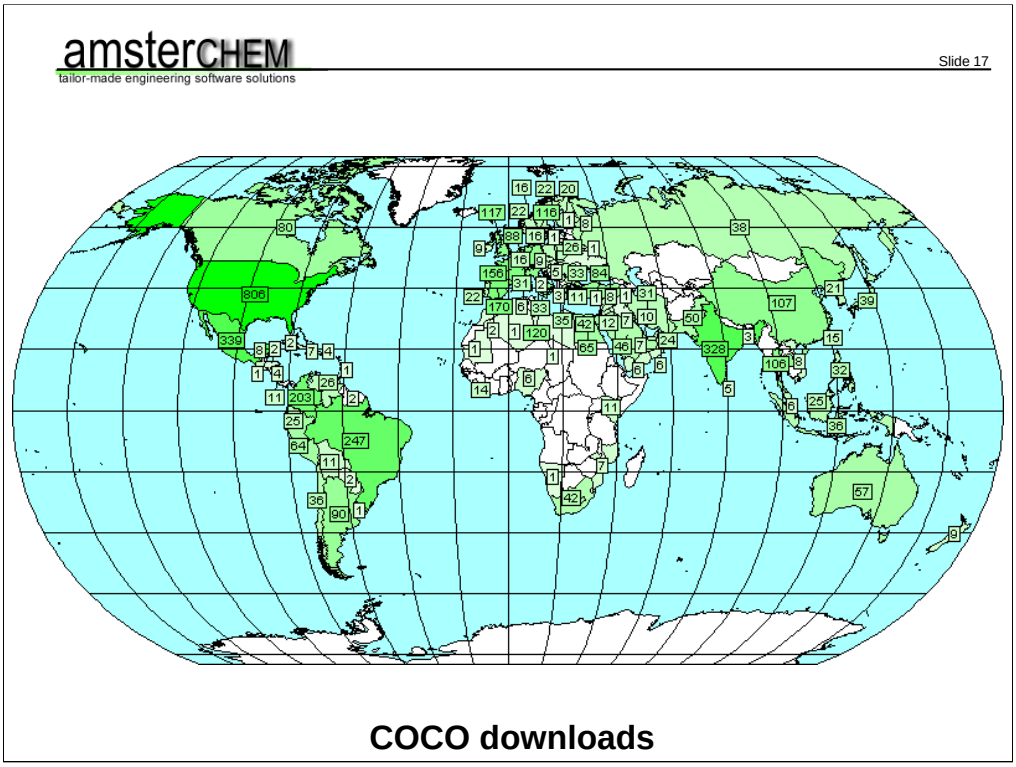
- Download COCO: http://www.cocosimulator.org/ (or ask for a copy during the workshop)
- Contact amster**CHEM** for CAPE-OPEN consulting
- Interoperability testing program: http://www.cocosimulator.org/index_compliancy.html

Acknowledgements:

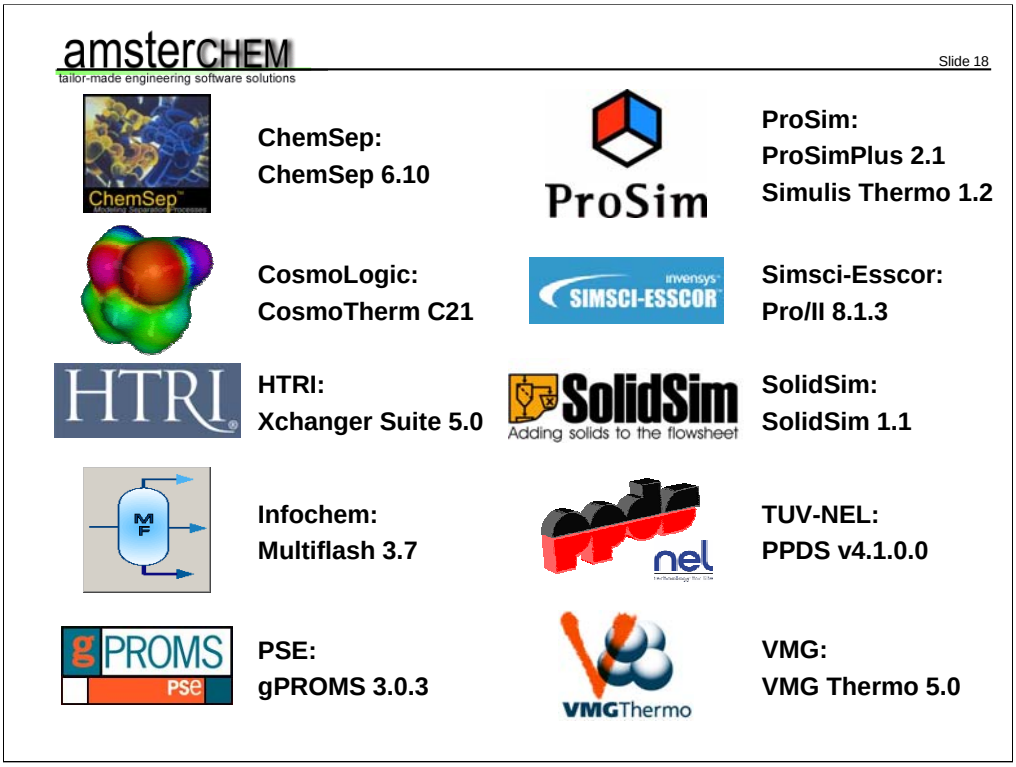
- Richard Baur
- ChemSep: Ross Taylor, Harry Kooijman
- Cosmo*THERM*: Frank Eckert
- Michel Pons

COCO is available for download from cocosimulator.org. If you want my help in CAPE-OPEN software issues, please contact me at jasper@amsterchem.com. As always I call on people to join the interoperability programme. If you have CAPE-OPEN software or components, please see the compliancy testing page at cocosimulator.org for more information.

I would like to thank Richard Baur, who could not be here today. I of course would like to thank Ross Taylor and Harry Kooijman for ChemSep and Frank Eckert for CosmoTherm. And of course we are all grateful for Michel's efforts.



COCO is being downloaded every day from all over the world. Feel free to download your own copy today.



COCO would not be what it is today without continuous interoperability testing. So I would much like to thank all the people that provide me with the licenses to do so. Thank you.