CAPE-OPEN

Delivering the power of component software
and open standard interfaces
in Computer-Aided Process Engineering

Open Interface Specification:
Physical Properties Data Bases Interface

www.colan.org

ARCHIVAL INFORMATION

Filename	Physical Properties Data Bases Interface Specification.doc
Authors	CO-LaN consortium
Status	Public
Date	August 2003
Version	version 2
Number of pages	132
Versioning	version 2, reviewed by Jean-Pierre Belaud, August 2003
	version 1, November 2001
Additional material	
Web location	www.colan.org
Implementation specifications version	CAPE-OPENv1-0-0.idl (CORBA) CAPE-OPENv1-0-0.zip and CAPE-OPENv1-0-0.tlb (COM)
Comments	

IMPORTANT NOTICES

SUMMARY

This paper defines a CAPE-OPEN compliant standard interface for connecting a data base with recorded physical property values and with model parameters to flowsheeting and other engineering programs.

This interface deals with physical property data at discrete values of the state variables (temperature, pressure, composition), as far as measured, correlated or estimated values are concerned. There will be no access methods that deliver recorded thermophysical property values exactly at a given state. However, parameters of model equations will be delivered that can be used for calculating data at any desired state.

This document defines the interface in textual form and both as COM and CORBA IDLs. It also contains a COM sample prototype.

ACKNOWLEDGEMENTS

CONTENTS

1. INTRODUCTION 9

2. REQUIREMENTS 10

2.1 TEXTUAL REQUIREMENTS 10
2.2 USE CASES 11
2.2.1 Actors 11
2.2.2 List of Use Cases 11
2.2.3 Use Cases maps 12
2.2.4 Use Cases 14
2.3 SEQUENCE DIAGRAMS 20

3. ANALYSIS AND DESIGN 21

3.1 OVERVIEW 21
3.2 SEQUENCE DIAGRAMS 21
3.3 INTERFACE DIAGRAMS 25
3.4 STATE DIAGRAMS 26
3.5 OTHER DIAGRAMS 27
3.6 INTERFACES DESCRIPTIONS 27
3.6.1 ICapePpdbRegister 29
3.6.2 ICapePpdb, ICapePpdbTables and ICapePpdbModels 31
3.6.3 ICapePpdb 33
3.6.4 ICapePpdbTables 46
3.6.5 ICapePpdbModels 58
3.6.6 ICapePpdbAbstractModel 72
3.6.7 Necessary extensions of CAPE-OPEN lists 75
3.7 SCENARIOS 76

4. INTERFACE SPECIFICATIONS 77

4.1 COM IDL 77
4.2 CORBA IDL 77

5. NOTES ON THE INTERFACE SPECIFICATIONS 78

6. PROTOTYPES IMPLEMENTATION 79

6.1 CLASS CAPEIDENTIFICATION 79
6.2 CLASS CAPEPPDBREGISTER 79
6.3 CLASS CAPEPPDB 80
6.4 CLASS PPDX 80
6.5 CLASS STRINGARRAY 80
6.6 CLASS PLISTEN 80
6.7 FUNCTIONS FOR MANIPULATING AUTOMATION TYPES 80
6.8 REPOSITORY 80
6.9 THE CLIENT 81
6.10 INSTALLATION 81
6.11 DESINSTALLATION 81

7. SPECIFIC GLOSSARY TERMS 82

8. BIBLIOGRAPHY 83

8.1 PROCESS SIMULATION REFERENCES 83
8.2 COMPUTING REFERENCES 83
8.3 GENERAL REFERENCES 83

9. APPENDICES 84

9.1 APPENDIX A: PROPERTIES 84
9.2 APPENDIX B: UNITS 97
9.3 APPENDIX C: MODELS OR EQUATIONS 124
9.4 APPENDIX D: PHASE EQUILIBRIUM INFORMATION 129
9.5 APPENDIX E: STATE OF AGGREGATION 130
9.6 APPENDIX F: TABLE INFORMATION, MODEL PARAMETER SET INFORMATION 131
9.7 APPENDIX G: ADDITIONAL PROPERTY SPECIFICATIONS 131

LIST OF FIGURES

Figure 1 Use cases for the PPDB administrator 12
Figure 2 Uses cases for the flowsheet user 13
Figure 3 Use cases for the thermo system 13
Figure 4 Use cases for the simulator 14
Figure 5 Use cases for a data regression system 14
Figure 6 Sequence diagram for PPDB administrator 21
Figure 7 Sequence diagram for flowsheet user 22
Figure 8 Sequence diagram for thermo system 23
Figure 9 Sequence diagram for a simulator 24
Figure 10 Sequence diagram for a data regression system 25
Figure 11 Class and interface diagram 26
Figure 12 State diagram for ICapePpdbRegister 26
Figure 13 State diagram for ICapePpdb 27

1. Introduction

Thermophysical property data are needed by chemical engineers, chemists or other workers in the chemical or process engineering industry for a variety of purposes:

- ❑ process development
- ❑ flowsheet simulation
- ❑ plant optimization
- ❑ production control

Since many years, computer programs have been a valuable tool for the chemical engineer. However, on the market today there is no single program package suitable for all purposes, but there are several programs available, which help in performing many of the engineer's tasks. In order to get consistent results, it is necessary to use exactly the same physical property data for all these programs and all these tasks.

In the past, each computer program used its own data base. So the consistency of data throughout the whole engineering process was not supported. Some programs allowed the user to specify its own data sets, but most of them used different formats. Therefore, data transfer between the programs could only be done by hand. This procedure is not only laborious but also prone to errors.

On the other hand, large commercial data bases containing thermophysical property data have evolved in the last two decades. They allow quick access to almost every value which has been published. However, the transfer to the engineering programs had to be done by a data-typist or the scientist in the old-fashioned way.

The aim of this document is to define a standard set of interfaces for connecting thermophysical property data bases with user programs, allowing the automatic exchange of property data and of model parameters. Such an interface consists of a set of type declarations and a set of methods for exchanging data between data bases and user programs. The terms "interface" and "method" stem from the object-oriented specification and programming. "Interface" means a class having no public variables (CORBA) or a subset of the public methods of a class (COM). A classical programmer may think of modules and functions (subroutines, procedures) instead of interfaces and methods.

The set of interfaces to be defined should be a standard one. That means, it has to be acceptable both by data base and user program producers. For a smooth data transfer, not only the method calls but also the transferred contents have to be standardized. That means, a PPDB server following this standard should transfer its data sets in a well-defined way. For instance, the values' units and the coding of their meaning will follow certain rules.

This document contains a set of methods for getting thermophysical property values from a variety of data bases, which can be used by a flowsheet simulator or any other user program. This set of methods is kept as simple as possible. Hopefully, this simplicity will allow many database producers and simulator vendors to implement it and to include it in their products.

2. Requirements

This chapter contains the requirements developed by the project team. It contains a textual description followed by a number of use cases.

2.1 Textual requirements

This interface specification deals with thermophysical property data and models taken from the literature. In order to distinguish it from the interfaces developed in the "CAPE-OPEN Thermo Specification" project[6,13], which deal with calculated property values at given states, the term "physical property data base" is introduced:

A **physical property data base (PPDB)** is an abstract model for all types of collections with thermophysical property data and with parameters of models for calculating thermophysical property data that have been taken from the literature, have been measured or processed in one's own laboratory or have been obtained from other sources. Such a data base can be implemented in a variety of physical manners. It can be

- ❑ a relational data base
- ❑ an online service
- ❑ a neutral file

Throughout this specification a PPDB is defined to consist of data tables, which have a header with the definition of the properties and their units and a body with the numerical values. There is no distinction between dependent and independent variables. Each table is linked to a mixture description and a set of bibliographic specifications.

A PPDB can contain any type of data. However, all properties must fit into a catalogue of properties. This catalogue has to be agreed on previously and should be treated as a CAPE-OPEN standard. It must also contain a list of calculation methods, for which model parameters are stored, and the exact form of their equation. This catalogue makes the whole standard very flexible, because additional properties can be added easily, and this means, that the standard will evolve.

In principle, a PPDB can contain any type of pure compounds and mixtures with any state of aggregation: organic, inorganic, plastics, materials (metals and alloys), emulsions. However, in the first stage of standardization, there should be a restriction to regular chemicals. A "regular chemical" should be defined as a substance that is listed in one of the registers of the Chemical Abstracts Service (Columbus, Oh, USA) and has a Chemical Abstracts Registry Number (CAS-number). Chemical Abstracts may be criticized that they sometime list a substance twice with different CAS-numbers, but it has the largest collection of chemical substances and other collections are as well erroneous. But the standard should also accept substances having no CAS-numbers. Especially technical mixtures like mineral oils, heat transfer fluids or victuals (food) have to be taken into account. They should be treated as "pseudo-compounds".

Because a PPDB can have any internal structure, there must exist a general interface for linking PPDBs to user programs, that is independent of the internal structure of the PPDB.

Steps of data transfer

The internal implementation of a PPDB interface-set may be complicated, because there are so many different types of PPDBs, but it is not of interest for a client. He needs to see only two objects with the following access methods.

- a catalogue of the available data bases

    - display contents of catalogue
- an object representing a PPDB
    - opening a data base
    - closing a data base
    - list of available compounds
    - list of dictionary information, e.g. available journals or stored properties
    - search for properties
    - obtain overview of tables found
    - fetch numerical data and experimental errors contained in found tables
    - fetch model parameter contained in found tables
    - fetch chemical mixture belonging to found tables
    - fetch bibliographic specification belonging to found tables

In order to keep this set of interfaces as simple as possible, model parameters are in many aspects treated as property data.

User programs have to different ways to use data stored in a PPDB:

- Direct retrieval of the data
- Retrieval of models or model parameters, which can later be used for calculating property values.

2.2 Use Cases

2.2.1 Actors

There are five principal actors who deal with the physical property database interface.

(i) The PPDB administrator is responsible for creating and maintaining databases.

(ii) The flowsheet user (human engineer) selects a data base, executes queries and loads data sets.

(iii) A thermo system is a program for calculating thermodynamic data.

(iv) A simulator is a program system for calculating the behavior of processes by assembling flowsheets form unit operations.

(v) A data regression system calculates smoothed values or parameters from experimental data.

2.2.2 List of Use Cases

- ❑ UC-3-1: Set up of the PPDB system

- UC-3-2: Installing a new PPDB
- UC-3-3: Browsing thru the repository of PPDBs
- UC-3-4: Opening a PPDB
- UC-3-5: Searching for data
- UC-3-6: Searching for models
- UC-3-7. Browsing thru table information
- UC 3-8: Browsing thru model parameter set information
- UC-3-9: Having data displayed
- UC 3-10: Load data
- UC-3-11: Calculate data
- UC 3-12: Load parameter sets

2.2.3 Use Cases maps

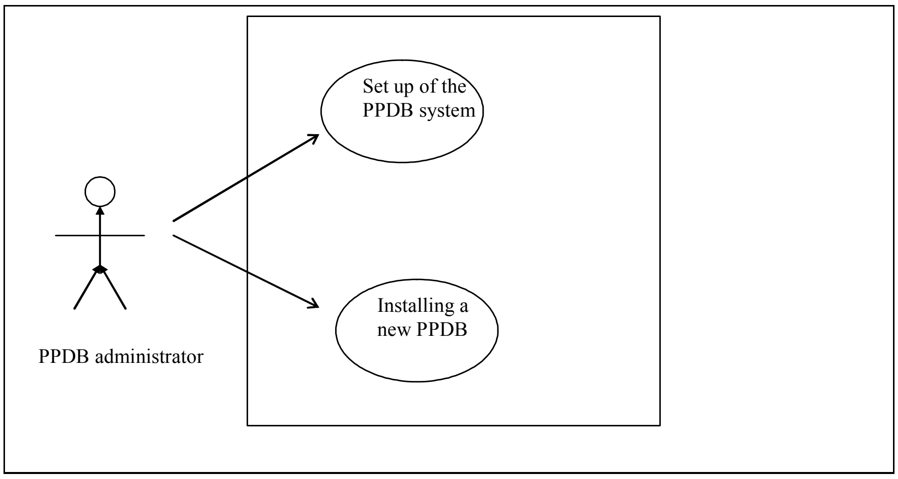
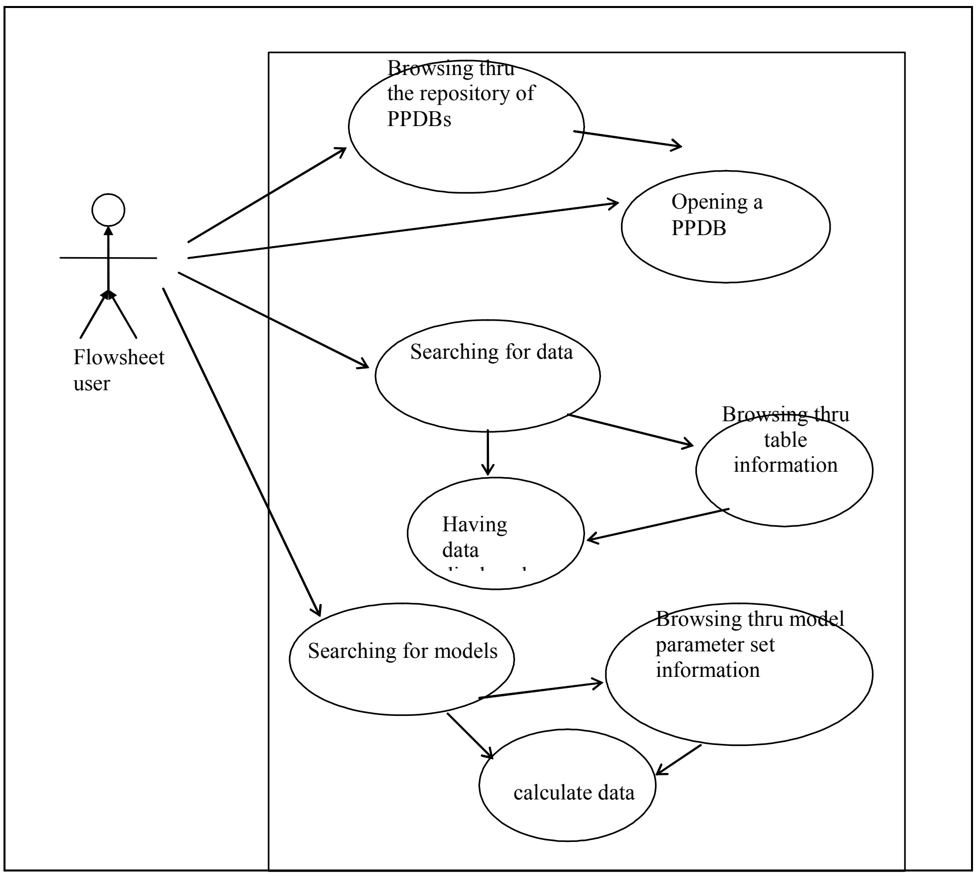


Figure 1 Use cases for the PPDB administrator



Browsing thru the repository of PPDBs
Opening a PPDB
Flowsheet user
Searching for data
Browsing thru table information
Having data
Searching for models
Browsing thru model parameter set information
calculate data

Figure 2 Uses cases for the flowsheet user

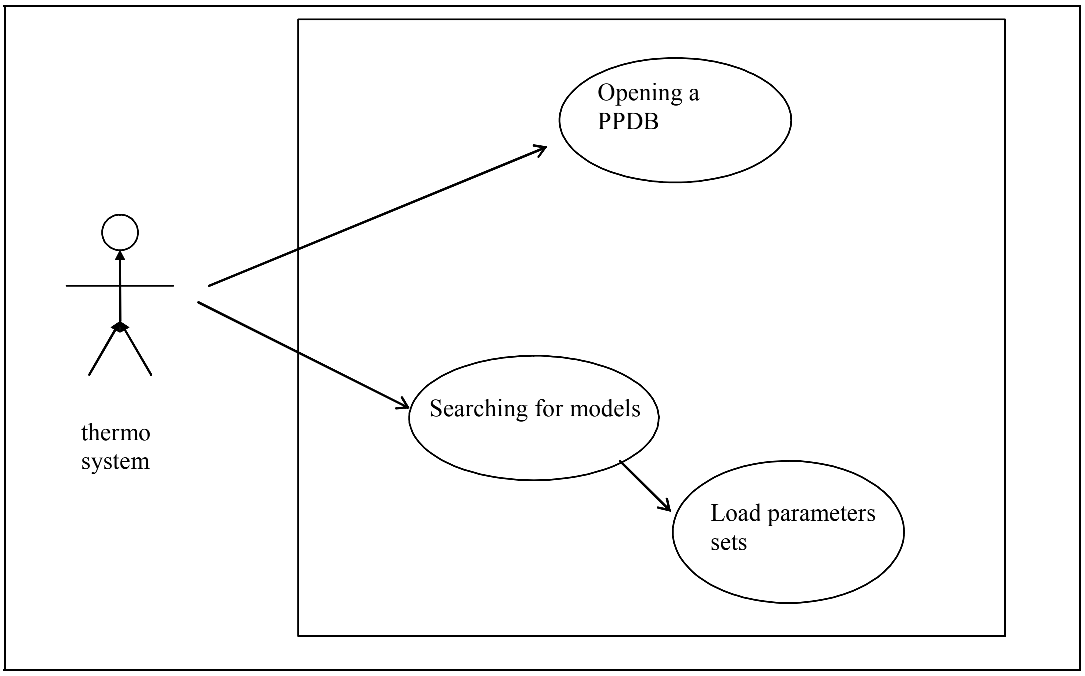


Figure 3 Use cases for the thermo system

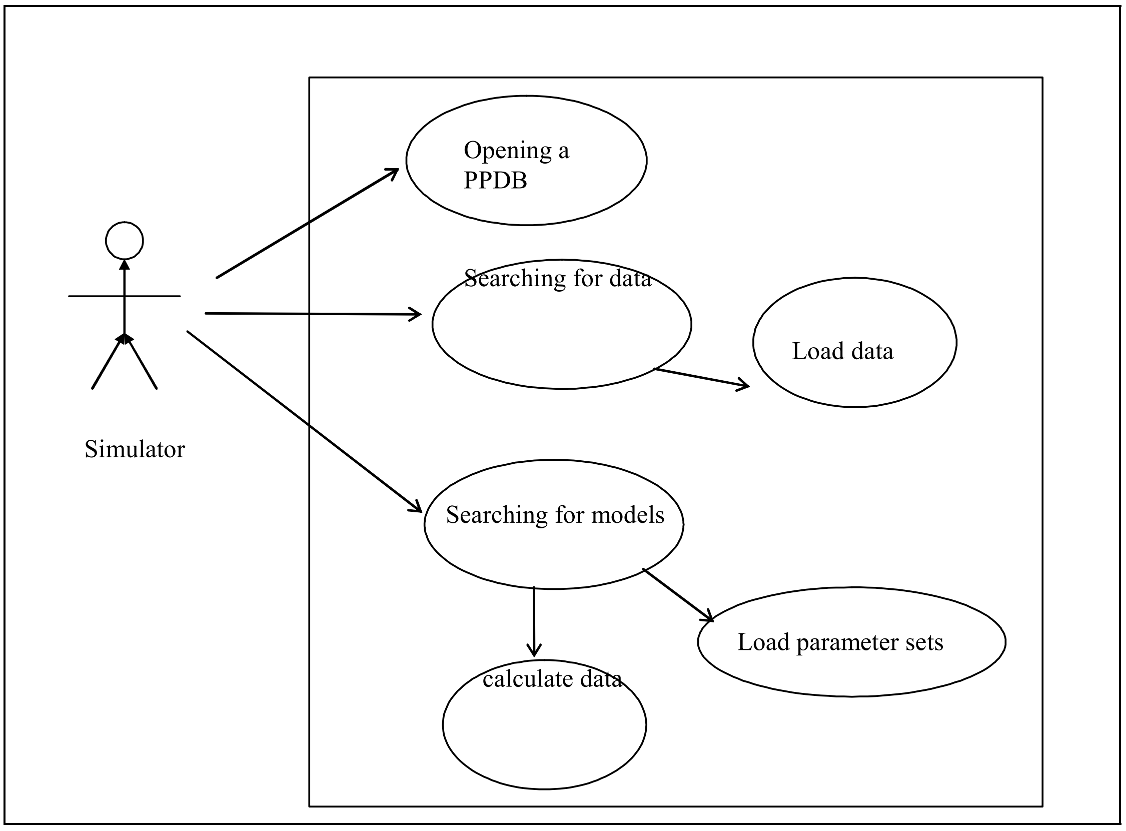


Figure 4 Use cases for the simulator

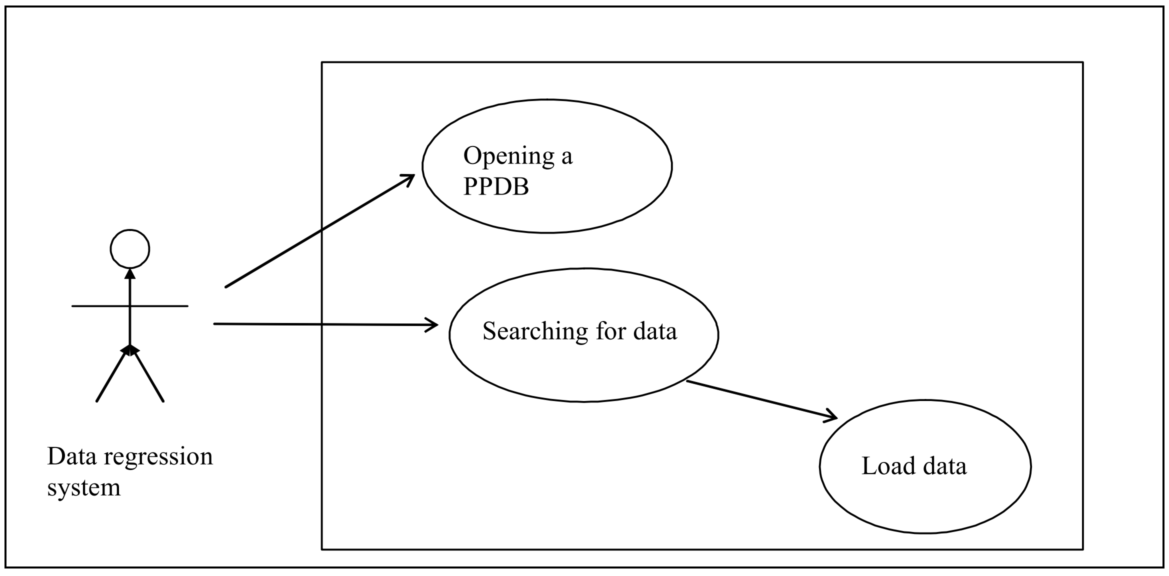


Figure 5 Use cases for a data regression system

2.2.4 Use Cases

UC-001 NAME

Actors: list of actors

Priority: high/medium/low

Classification: categories to which the use case belongs

Context: context information needed to understand the flow of events

Pre-conditions: anything required before starting the flow of events

Flow of events: description of basic path and some alternatives

Post-conditions: conditions that will be satisfied after completion of the flow of events

Errors: list of exceptions possibly during execution of the use case

Uses: subordinate use cases (other use cases utilized by this one)

Extends: generic use case from which this one is derived

UC-3-1: SET UP OF THE PPDB SYSTEM

Actors: PPDB administrator

Description: The PPDB administrator installs a repository of PPDBs and makes its location available to all users.

Priority: high

Classification: installation

Context:

Pre-conditions:

Flow of events:

Post-conditions: New PPDBs can be installed now.

Exceptions:

Uses:

Extends:

Note: This use case only deals with functions of the server. It need not to be taken care of in this definition.

Proposed implementation: In MS-Windows environments, the repository is managed by the key HKEY_LOCAL_MACHINE/SOFTWARE/CAPE/PPDB/register of the Windows registry.

UC-3-2: INSTALLING A NEW PPDB

Actors: PPDB administrator

Description: The PPDB administrator obtains a new neutral file or a whole database from an external person. He stores the data on his central server and puts the name of the new data collection and its location into the repository of PPDBs.

Priority: high

Classification: installation

Context:

Pre-conditions: Implementation of PPDB repository must exist

Flow of events:

Post-conditions: PPDB is ready for use.

Exceptions: PPDB repository not available

Uses:

Extends:

Note: This use case only deals with functions of the server. It need not to be taken care of in this definition.

Proposed implementation: In MS-Windows environments, the PPDB administrator needs only to install the new PPDB system on the server and to extend the key HKEY_LOCAL_MACHINE/SOFTWARE/CAPE/PPDB/register by new entries with name = *PPDB name*, value = *access information|internal Parameters.*

UC-3-3: BROWSING THRU THE REPOSITORY OF PPDBS

Actors: flowsheet user

Description: The flowsheet user establishes a connection to the PPDB repository and has the list of all available PPDBs and a short description of their contents displayed. He continues with [opening a PPDB].

Priority: low

Classification: retrieval

Context:

Pre-conditions: PPDB repository exists.

Flow of events: Information on data bases is sent to the flowsheet user

Post-conditions:

Exceptions: PPDB repository not available

Uses: PPDB repository

Extends:

UC-3-4: OPENING A PPDB

Actors: flowsheet user, thermo system, simulator, data regression system

Description: A connection to a PPDB is established. Some data bases demand the specification of login information.

Priority: high

Classification: retrieval

Context:

Pre-conditions: PPDB repository exists. The name of the PPDB and - if necessary - its login information is known.

Flow of events: PPDB signals that it is open.

Post-conditions: PPDB is open. Queries can be submitted.

Exceptions: Failure in opening a data base, data base is already open

Uses:

Extends:

UC-3-5: SEARCHING FOR DATA

Actors: flowsheet user, simulator, data regression system

Description: A query for a certain data table or for all tables is specified and sent to the PPDB If the query was successful, i.e. if data have been found, the use cases [browsing thru table information], [having data displayed]or [load data] may follow.

Priority: high

Classification: retrieval

Context:

Pre-conditions: PPDB opened.

Flow of events: none

Post-conditions: The system changes its state according to the answer set found

Exceptions: Too many tables found, wrong arguments of method call, special type of query is not supported

Uses:

Extends:

UC-3-6: SEARCHING FOR MODELS

Actors: flowsheet user, thermo system, simulator

Description: A query for a certain model or for all models is specified and sent to the PPDB. If the query was successful, i.e. if data have been found, the use cases [Browsing thru model parameter set information], [Calculate data], [Calculate and load data] or [Load parameter sets] may follow.

Priority: high

Classification: retrieval

Context:

Pre-conditions: PPDB opened.

Flow of events: none

Post-conditions: The system changes its state according to the answer set found

Exceptions: Too many tables found, wrong arguments of method call. special type of query is not supported

Uses:

Extends:

UC-3-7. BROWSING THRU TABLE INFORMATION

Actors: flowsheet user

Description: The user requests for information on the tables found in a former search. He may continue with [having data displayed]

Priority: high

Classification: retrieval

Context:

Pre-conditions: [Searching for data]

Flow of events: headings of the tables found

Post-conditions: The system changes its state, in order to be ready for displaying full information on the tables found

Exceptions: Last table was already displayed; no successful query was performed before.

Uses:

Extends:

UC-3-8. BROWSING THRU MODEL PARAMETER SET INFORMATION

Actors: flowsheet user

Description: The user requests for information on the tables found in a former search. He may continue with [having data displayed].

Priority: high

Classification: retrieval

Context:

Pre-conditions: [Searching for models]

Flow of events: headings of the model parameter sets found

Post-conditions: The system changes its state, in order to be ready for displaying full information on the models found

Exceptions: Last table was already displayed, no successful query was done before.

Uses:

Extends:

UC-3-9: HAVING DATA DISPLAYED

Actors: flowsheet user

Description: The user requests the values of one of the tables found in the preceding search. He may also procure information on the chemical system and on the bibliography.

Priority: high

Classification: retrieval

Context:

Pre-conditions: [Searching for data], data were found

Flow of events: Data are sent to the flowsheet user.

Post-conditions: The system changes its internal state.

Exceptions: No successful query has been done before, last table is already displayed

Uses:

Extends:

UC-3-10: LOAD DATA

Actors: simulator, data regression system

Description: The values of one of the tables found in the preceding search are requested. They are accompanied by information on the chemical system and on the bibliography. All data are loaded by the calling program for further processing.

Priority: high

Classification: retrieval

Context:

Pre-conditions: [Searching for data], data were found

Flow of events: Data are sent to the requestor.

Post-conditions: The system changes its internal state.

Exceptions: No successful query has been done before, last table is already put out.

Uses:

Extends:

UC-3-11: CALCULATE DATA

Actors: flowsheet user, simulator

Description: The user requests that values are to be calculated from one of the model parameter sets found in the preceding search.

Priority: high

Classification: retrieval

Context:

Pre-conditions: [Searching for models], parameter sets were found

Flow of events: Calculated data are sent to the flowsheet user.

Post-conditions: The system changes its internal state.

Exceptions: No successful query has been done before, last model parameter set is already used

Uses:

Extends:

UC-3-12: LOAD PARAMETER SETS

Actors: thermo system, simulator

Description: The user requests the parameters of one of the model parameter sets found in the preceding search.

Priority: high

Classification: retrieval

Context:

Pre-conditions: [Searching for models], data were found

Flow of events: Data are sent to the calling program.

Post-conditions: The system changes its internal state.

Exceptions: No successful query has been done before, last model parameter set is already worked off

Uses:

Extends:

2.3 Sequence diagrams

None.

3. Analysis and design

This chapter introduces the design. It contains a textual description followed by UML diagrams and a detailed description of the interface.

3.1 Overview

The interface deals with reading discrete data - either experimental or produced by smoothing or calculating - and model parameters from a PPDB. The client should not be aware of the actual type of the PPDB, but all PPDBs should support the same kind of data.

Data are organized as tables. This should not be misunderstood as a table of a relational data base. A table consists of numerical values, grouped in data points - one or more values that belong together, e.g. temperature, pressure, viscosity - and a table header which defines the properties and the units. Tables are linked to a definition of a pure compound or a mixture and a bibliographic specification.

Model parameters are parameters for an equation that allow a property to be calculated. They are organized as parameters sets. Such a parameter set consists of the parameters itself and their range of validity. Like a table it is linked a definition of a pure compound or a mixture and a bibliographic specification.

3.2 Sequence diagrams

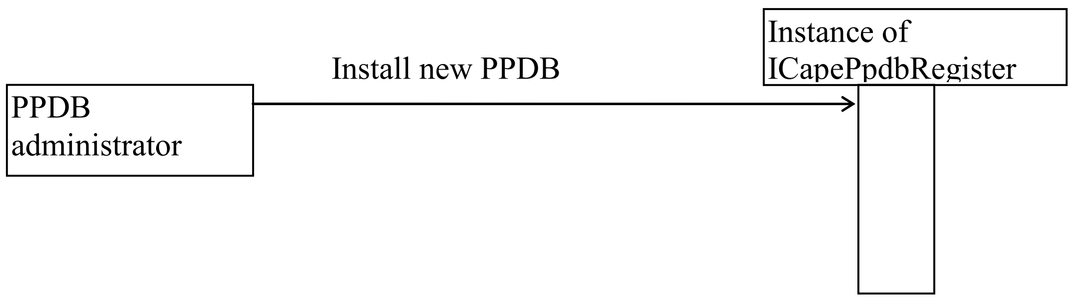


Figure 6 Sequence diagram for PPDB administrator

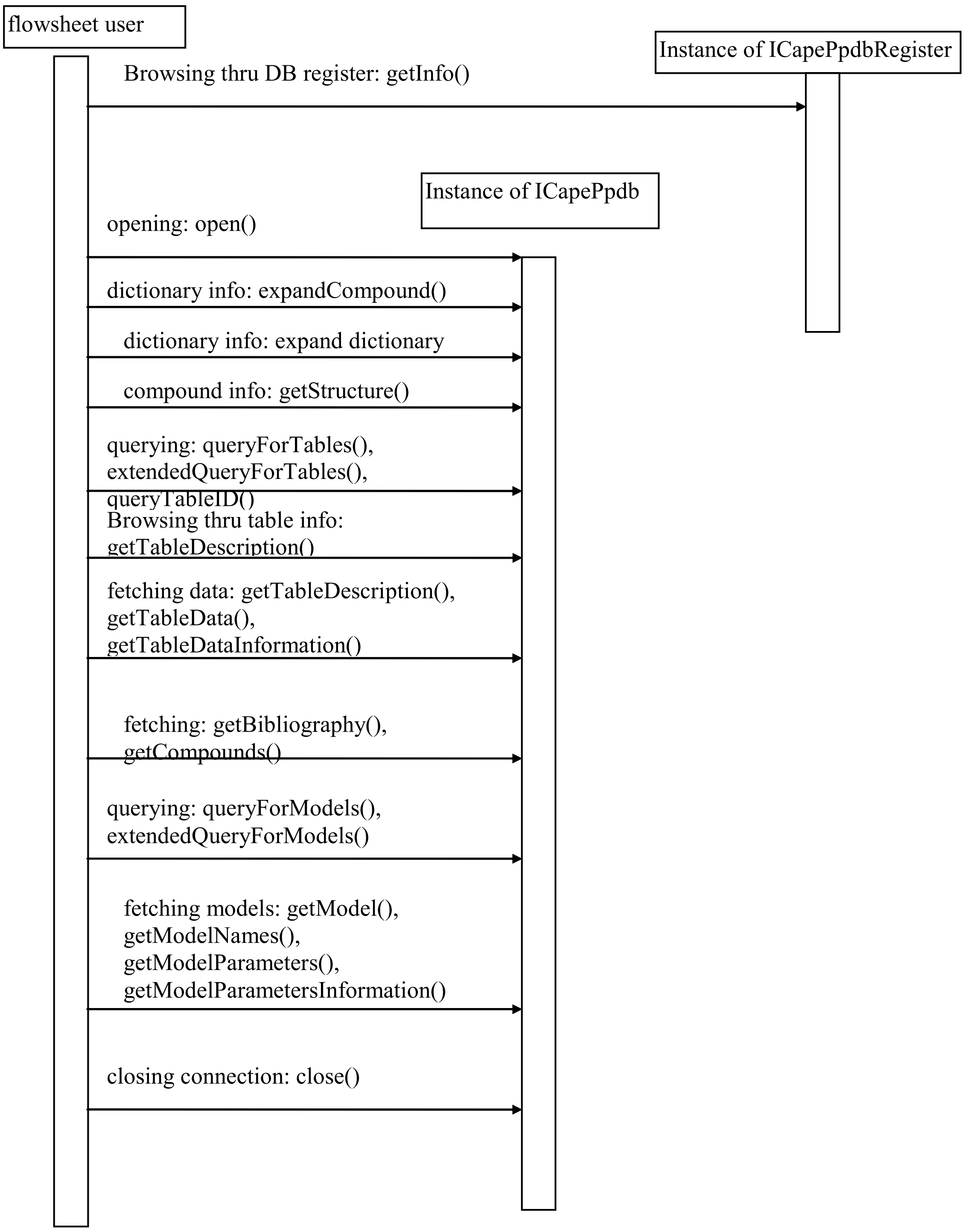

flowsheet user
Instance of ICapePpdbRegister
Browsing thru DB register: getInfo()
Instance of ICapePpdb
opening: open()
dictionary info: expandCompound()
dictionary info: expand dictionary
compound info: getStructure()
querying: queryForTables(), extendedQueryForTables(), queryTableID()
Browsing thru table info: getTableDescription()
fetching data: getTableDescription(), getTableData(), getTableDataInformation()
fetching: getBibliography(), getCompounds()
querying: queryForModels(), extendedQueryForModels()
fetching models: getModel(), getModelNames(), getModelParameters(), getModelParametersInformation()
closing connection: close()

Figure 7 Sequence diagram for flowsheet user

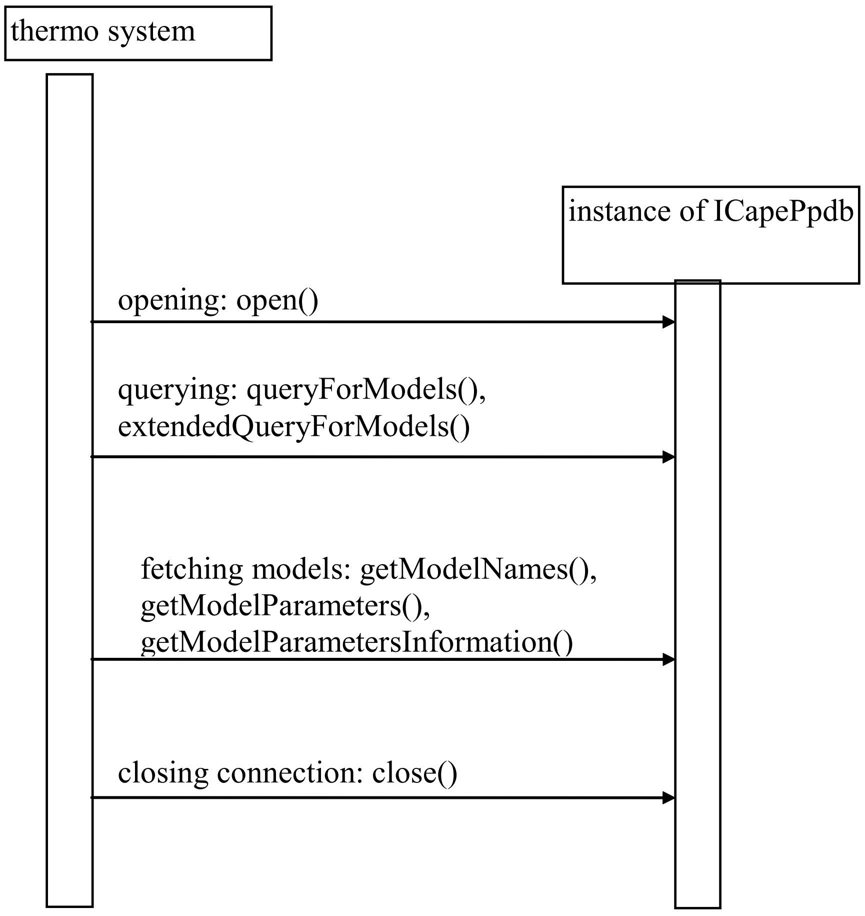

thermo system
instance of ICapePpdb
opening: open()
querying: queryForModels(), extendedQueryForModels()
fetching models: getModelNames(), getModelParameters(), getModelParametersInformation()
closing connection: close()

Figure 8 Sequence diagram for thermo system

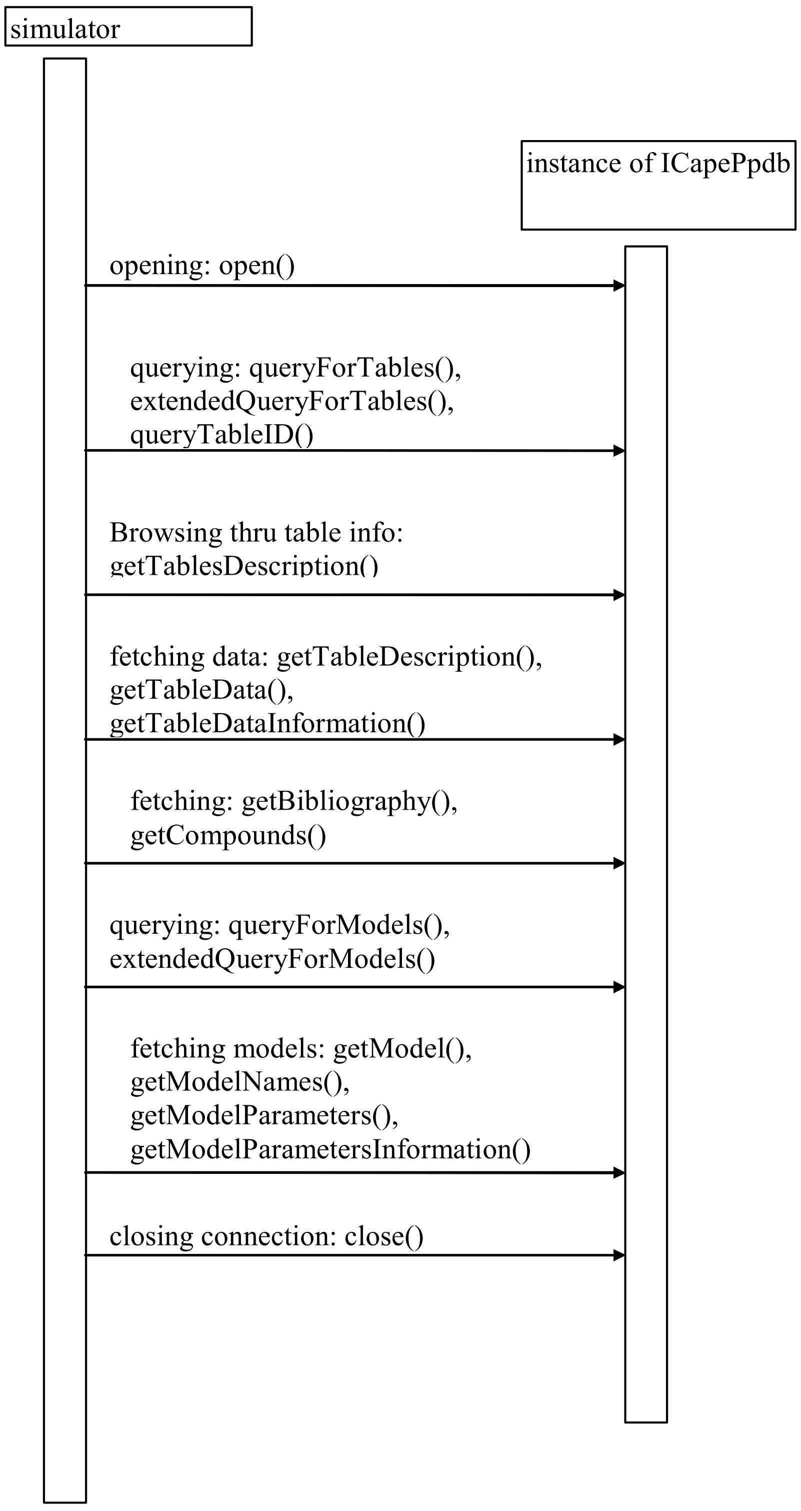

simulator
instance of ICapePpdb
opening: open()
querying: queryForTables(), extendedQueryForTables(), queryTableID()
Browsing thru table info: getTablesDescription()
fetching data: getTableDescription(), getTableData(), getTableDataInformation()
fetching: getBibliography(), getCompounds()
querying: queryForModels(), extendedQueryForModels()
fetching models: getModel(), getModelNames(), getModelParameters(), getModelParametersInformation()
closing connection: close()

Figure 9 Sequence diagram for a simulator

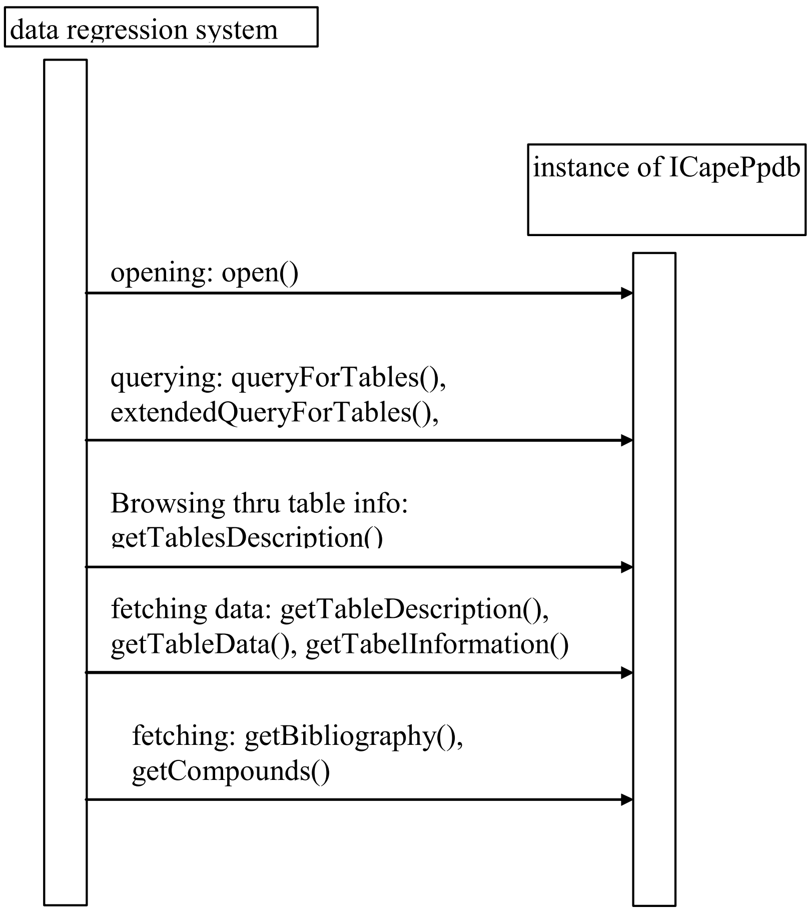

data regression system
instance of ICapePpdb
opening: open()
querying: queryForTables(), extendedQueryForTables(),
Browsing thru table info: getTablesDescription()
fetching data: getTableDescription(), getTableData(), getTabelInformation()
fetching: getBibliography(), getCompounds()

Figure 10 Sequence diagram for a data regression system

3.3 Interface diagrams

<u>IN- 3-1</u>

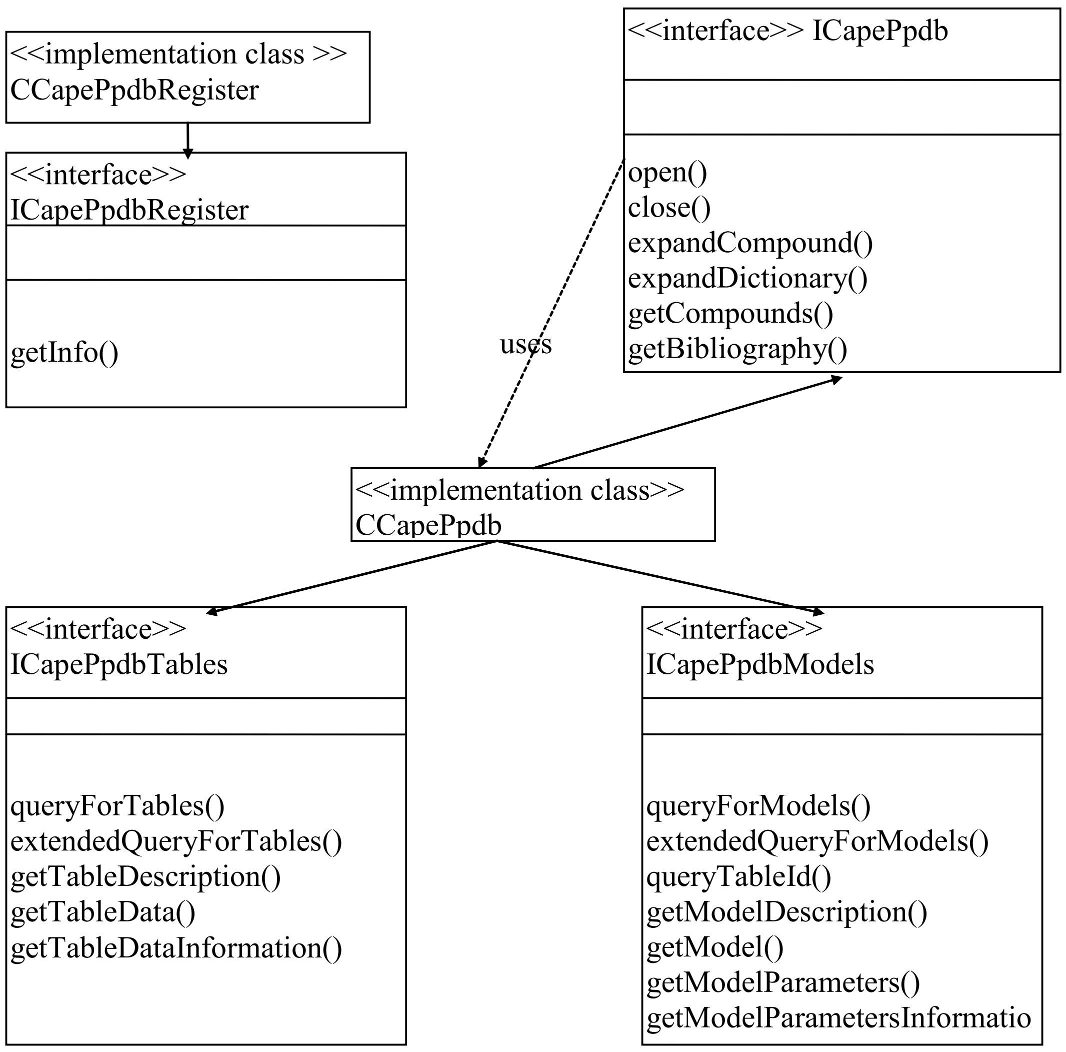


Figure 11 Class and interface diagram

3.4 State diagrams

ST-3-1: ICAPEPPDBREGISTER

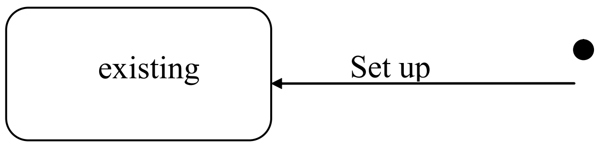


Figure 12 State diagram for ICapePpdbRegister

ST-3-2: ICAPEPPDB

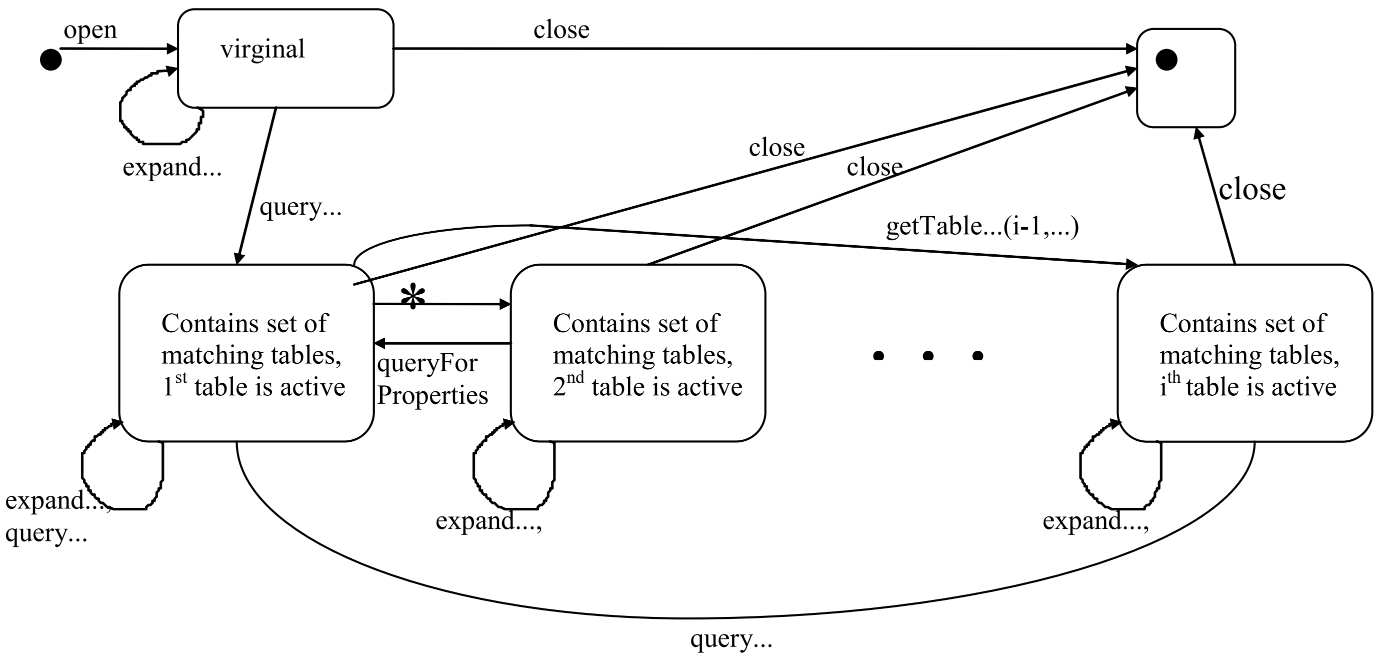

open
virginal
close
expand...
query...
close
close
close
getTable...(i-1,...)
*
Contains set of matching tables, 1st table is active
queryFor Properties
Contains set of matching tables, 2nd table is active
Contains set of matching tables, ith table is active
expand..., query...
expand...,
expand...,
query...

Figure 13 State diagram for ICapePpdb

*For each of the method calls GetTableDescription(...),GetTableData(0,...), GetTableDataInformation (0,...), GetModel(0,...), GetModelParameters(0,...), GetModelParametersInformation(0,...), GetCompounds(0,...) and GetBibliography(0,...) there has to be a set of n (n is the number of tables found) such states, so totally there are n^8 states.

The state of the server changes after each query and after each transfer of information. The number of states is very large, but limited. So, the server is a finite automaton. But, since the number of states is so large, the state diagram does not help very much in the design process.

3.5 Other diagrams

None.

3.6 Interfaces descriptions

The CO common base types are used[14] such as CapeShort, CapeFloat, …

A special value is needed for each type to express that a property value is missing. It is not sufficient to use 0 (zero), because 0 may be a reasonable property value, e.g. for dipole moments. These numerical values can only be used for designating UNDEFINED. They are defined within the CO common base types such as CapeLongUNDEFINED, CapeStringUNDEFINED, …

The interface uses some symbolic constants. Their values should be fixed, in order to have a minimum of machine dependence.

- For the specification of compounds for Expand... and Query... methods

symbolic constant	value	compound is identified by
CAPE_SpecCompoundFormula	0	by its sum formula
CAPE_SpecCompoundName	1	by its name (synonym or systematic or internal name)
CAPE_SpecCompoundCasNo	2	by its CAS number
CAPE_SpecCompoundDbSpecificId	3	by its data base specific ID
CAPE_SpecCompoundStructure	4	by its structure
CAPE_SpecCompoundDescriptor	5	by its descriptor (family)

❑ For specifying the dictionary type, when "ExpandDictionary" is called:

symbolic constant	value	meaning
Cape_DicPropertyList	0	dictionary of all properties (cf. Appendix A)
Cape_DicUnits	1	dictionary of all units (cf. Appendix B)
Cape_DicUnitsTemperature	2	dictionary of all temperature units
Cape_DicUnitsPressure	3	dictionary of all pressure units
Cape_DicModels	4	dictionary of all model equations that are supported by the PPDB (cf. Appendix C)
Cape_DicPhaseEq	5	dictionary of all phase equilibrium information (cf. Appendix D)
Cape_DicPhases	6	dictionary of all valid states of aggregation (cf. Appendix E)
Cape_DicTableInformation	7	dictionary of all table information phrases (cf. Appendix F)
Cape_DicPropInformation	8	dictionary of all property information phrases (cf. Appendix G)
Cape_DicSetInformation	9	dictionary of all model parameter set information phrases (cf. App. F)
Cape_DicCompoundDescriptors	10	dictionary of all compounbd descriptors (family names)

❑ The type of a chemical structure formula (method GetStructure) is specified by

symbolic constant	value	meaning
Cape_SpecStructureDDB	0	DDB-Artist format[2]
Cape_SpecStructureMDL	1	MDL format[3]
Cape_SpecStructureSMILES	2	SMILES format[4]

Chemical substances are characterized by the elements of this structure "CapePpdbCompound". If some of its attributes are not supported by a PPDB, they may be left UNDEFINED (CapeString UNDEFINED or CapeArrayStringUNDEFINED), but at least either "name", "casNo" or "dbSpecifcId" is needed.

A sequence/array of CapePpdbCompound is defined as “CapeArrayPpdbCompound”.

Type and name of structure element	**meaning**
CapeString name	systematic name (unique for PPDB)
CapeString sumFormula	sum formula in upper and lower case letters As a result of a query, the formula is delivered in Hill nomenclature[1] (organic compounds: first C, then H, other atoms alphabetical; inorganic compounds: all atoms alphabetical),but for a query ("ExpandFormula"), the atoms may be in any order. For complex cases like hydrohalogenides, hydrates or addition compounds the sum formula may either be left CapeStringUNDEFINED or follow the following rule: <sum formula of main compound>.<sum formula of additional compound> e.g. Mg2O4S.6H2O, C6H7N.ClHF.
CapeString casNo	Chemical Abstracts Registry number
CapeString dbSpecificId	a compound identification that is specific for the open PPDB (mainly used if the CAS number is not available)
CapeString internalName	e.g. name of compound used in company
CapeArrayString synonyms	a set of synonyms used for the compounds
CapeArrayString descriptors	a set of descriptors for the compounds, e.g. family names used by DIPPR

For implementation of this structure in DCOM see chapter 5.

Error handling

Errors are signaled by an exception (CORBA) or a HRESULT error code, that has been standardized by the CAPE-OPEN group[5].

Error Code	**meaning**
S_OK (only COM)	Method call was successful.
ECapeData	error in the internal data
ECapePersistenceNotFound	A data bases or an item of a data bases was not found.
ECapeNoMemory	The resources of the computer were exhausted, allocation error.
ECapeUnknown	internal error
ECapeBadArgument	The argument of the method call was wrong.
ECapeNoImpl	Called method is not implemented.
ECapeLimitedImpl	Some limits of the current implementation were exceeded.
ECapeInvalidOperation	This operation is not valid in the current context.

The exact meanings of the error codes can be found in the descriptions of the methods

3.6.1 ICapePpdbRegister

This Register knows all about the different types of PPDBs, which are accessible, and makes this knowledge available to the clients. It is managed by the system administrator, not the client users. It has only one method.

Interface Name	ICapePpdbRegister
Method Name	GetInfo
Returns	--

Description

This method is called by a client. It delivers a list of all accessible PPDBs. It returns ECapePersistenceNotFound, if there is no PPDB present.

Note: This interface should be implemented by a vendor-independent organization, because PPDBs from different vendors are to be included.However, at sites where only one type of PPDBs is used, a simple simplementation that deals only with this type of PPDB will do.

Arguments

Name	Type	Description
[out] ppdbNames	CapeArrayString	names of PPDBs
[out] ppdbInformations	CapeArrayString	Each element of this array contains information of a PPDB, depending on its type and implementation. If there are several types of information for a PPDB, they are separated from each other by commas. (i) neutral file format: - short description of the contents of the neutral file (ii) internal or external data base (May be a relational data base, a classical online service or a data collection accessible via HTTP) - short description of the contents of the data base - pricing information

Errors

ECapePersistenceNotFound: Register does not exist or is undefined.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

Interface Name	ICapePpdbRegister
Method Name	CreatePpdbObject (CORBA only)
Returns	CapeInterface (ICapePpdb)

Description

This method is only used with CORBA. It creates an instance of the class that implements the interfaces ICapePpdb, ICapePpdbTables and ICapePpdbModels and returns its address to the client.

Arguments

None.

Errors

ECapeUnknown: internal error

ECapeNoMemory: allocation error

3.6.2 ICapePpdb, ICapePpdbTables and ICapePpdbModels

These interfaces implement a physical property data base (PPDB). Such a PPDB may be a relational data base. The result of a query to such a data base is an answer-**set**. Since the popular programming languages cannot deal with sets, the elements of this answer-sets are fetched one by one.

The interfaces' methods can be arranged in five groups.

group name	**methods**	**description**
opening	Open	A database is opened and space for all static variables is reserved. Several PPDBs can be open at a time. After a PPDB has been opened, the client is informed, if the data base supports chemical structures search for chemical structures and substructures searching for compositions combining a search for a mixture with a search for all submixtures and compounds conversion of values to SI-units
closing	Close	An open PPDB is closed. All storage will be released.
expanding	ExpandCompound ExpandDictionary GetStructure	These methods inform the user on the contents of special dictionaries or lists, which can be used in searches.
searching	QueryForTables ExtendedQueryForTables	These methods perform a search. The result of such a search is an answer-set, not a single answer. The elements of this set can be obtained by the methods of the group "fetching".

	QueryForModels ExtendedQueryForModels QueryTableID	
fetching	GetTableDescription GetTableData GetTableDataInformation GetModelDescription GetModel GetModelParameters GetModelParametersInformation GetCompounds GetBibliography	These methods fetch the wanted information. They can only be called, after a method of the group "searching" has been executed successfully.

There are three methods that help the user to get information on the stored substances or other dictionary entries:

- **ExpandCompound:** Lists all substances that match a given set of specifications
- **ExpandDictionary**: Lists all property names, units, model names and other information that is available in the data bank.
- **GetStructure**: Retrieves the chemical structure of a given compound.

A data retrieval is done by invoking "**QueryForTables**", "**ExtendedQueryForTables, QueryForModels, "ExtendedQueryForModels"** or "**QueryTableID**". The data contained in the tables found as a result of a query can be obtained in two ways:

- Either a description of the tables found can be fetched by calling "**GetTableDescription**" and then a specific table can be obtained with "**GetTableData**" by specifying the rank number transferred by "GetTableDescription"
- or all tables found by "Query..." can be read sequentially by calling "**GetTableData**" with 0 as a rank number.

Wholly there are three methods for obtaining retrieved results after having invoked "QueryForTables" or "ExtendedQueryForTables":

- **GetTableDescription**: a short description of the tables found
- **GetTableData**: a data table with property definitions and values
- **GetTableDataInformation**: additional information on a data table

Four other methods can be used after a successful "QueryForModels" or "ExtendedQueryForModels".

- **GetModel**: Get an instance of a model equation with parameters:
- **GetModelNames:** names of the models that are contained in the documents found
- **GetModelParameters**: a set of model parameters

- **GetModelParametersInformation:** additional information on a set of model parameters

There are two methods that can be used after all types of queries.

- **GetCompounds** : definition of the chemical mixture or the pure compound, which belongs to the table
- **GetBibibiography**: bibliographic specification of the paper or book the table has been published in

The list given above shows that there are three types of methods.

(i) methods dealing with table data, i.e. measured or computed discrete data

(ii) methods taking care of model, i.e. parameters sets computed from experimental data

(iii) methods for both tables and models

type of methods	**methods**
methods for tables	QueryForTables, ExtendedQueryForTables, GetTableDescription, GetTableData, GetTableDataInformation
methods for models	QueryForModels, ExtendedQueryForModels, QueryTableID, GetModelDescription, GetModel, GetModelParameters, GetModelParametersInformation
methods for both tables and models	ExpandCompound, ExpandDictionary, GetStructure, GetCompounds, GetBibliography

In order to have not too many methods in one interface and to have a clear separation between tables and models, three interfaces where set-up:

(i)	ICapePpdb	methods, that can be used both for models and for tables
(ii)	ICapePpdbTables	methods which are only useful for plain data tables
(iii)	ICapePpdbModels	methods that can only be used for model parameters

3.6.3 ICapePpdb

Interface Name	ICapePpdb
Method Name	Open
Returns	--

Description

This method establishes a connection to a PPDB and opens it for further processing.

Arguments

Name	Type	Description
[in] ppdbName	CapeString	name of PPDB
[in] loginInformation	CapeString	Login information such as user name and password used by the database server. The exact format of this string depends on the PPDB, but for an ORACLE base data base it should look like *"user name/password"*. CapeStringUNDEFINED, if Ppdb does not need any login information
[out] isImplemented	CapeArrayBoolean	This boolean array indicates, which special functions are supported (corresponding array element = TRUE) or are not supported by the PPDB [0]: TRUE, if interface supports output of chemical structures [1]: TRUE, if interface supports search for fully defined chemical structures [2]: TRUE, if interface supports search for chemical substructures (partial defined structures) [3]: TRUE, if interface supports query for compositions [4]: TRUE, if interface supports queries for submixtures, i.e. search for a mixture and for all submixtures and pure compounds, the mixtures is consisting of. The search terms are combined by "OR". Example: submixture query for acetone/water/methanol means search for acetone/water/methanol OR acetone/water OR acetone/methanol OR water/methanol OR acetone OR water OR methanol [5]: TRUE, if interface can convert property values to SI-units [6]: TRUE, if interface supports dictionary expands [7]: TRUE, if interface supports expands of chemical compounds [8]: TRUE, if interface supports queries for and display of model parameters [9]: TRUE, if interface can deliver instances of ICapePpdbAbstractModel [10]: TRUE, if interface supports display of the original experimental data, from which model parameters have been

		created. [11]: TRUE, if interface supports a query for model names [12]: TRUE, if interface supports a query for a special phase equilibrium [13]: TRUE, if interface supports a query for a state of aggregation [14]: TRUE, if interface supports a query for a table information [15]: TRUE, if interface supports a query for additional property information [16]: TRUE, if interface supports a query for a range of temperature [17]: TRUE, if interface supports a query for a range of pressure

Errors

ECapePersistenceNotFound: Data base does not exist or is undefined

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapeData: Data bases contains data that cannot be processed by the server

ECapeBadArgument: Illegal login information given

Interface Name	ICapePpdb
Method Name	Close
Returns	--

Description

Closes the connection to a PPDB and deletes all associated variables.

Arguments

None.

Errors

ECapeUnknown: internal error

Interface Name	ICapePpdb
Method Name	ExpandDictionary
Returns	--

Description

Delivers the contents of a given dictionary. These are lists of items which the open PPDB may contain. This method needs not to be implemented. In this case, the argument isImplemented[6] of "open" has to be false.

Arguments

Name	Type	Description
[in] specType	CapeShort	Type of the dictionary the entries of which are wanted as symbolic constant Cape_Dic... Cape_DicPropertyList (0) dictionary of all properties Cape_DicUnits (1) dictionary of all units Cape_DicUnitsTemperature (2) dictionary of all temperature units Cape_DicUnitsPressure (3) dictionary of all pressure units Cape_DicModels (4) dictionary of all model equations Cape_DicPhaseEq (5) dictionary of all phase equilibrium information Cape_DicPhases (6) dictionary of all valid phases Cape_DicTableInformation (7) dictionary of all table information phrases Cape_DicPropInformation (8) dictionary of all property information phrases Cape_DicSetInformation (9) dictionary of all set information phrases Cape_DicCompoundDescriptors (10) dictionary of all compound descriptors (family names)

[in] item	CapeString	A string that dictionary items shall match. It may contain wildcards. ? exactly one character * a string of zero or more characters CapeStringUNDEFINED: The whole contents of the dictionary should be transferred
[out] dictionaryEntries	CapeArrayString	contents of the selected dictionary (that match "item")

Errors

ECapePersistenceNotFound: Specified dictionary does not exist

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

Interface Name	ICapePpdb
Method Name	ExpandCompound
Returns	--

Description

Delivers a list of compounds that match a set of given specifications. If several specifications are given, they must all match.

This method needs not to be implemented. In this case, the argument isImplemented[6] of "open" has to be false.

Arguments

Name	Type	Description
[in] types	CapeArrayShort	types of the given specifications (next parameter) as symbolic constant SPEC... CAPE_SpecCompoundFormula (0) by its sum formula CAPE_SpecCompoundName (1) by its name (synonym or systematic or internal name) CAPE_SpecCompoundCasNo (2) by its CAS number CAPE_SpecCompoundDbSpecificId (3) by its data base specific ID CAPE_SpecCompoundStructure (4) by its structure CAPE_SpecCompoundDescriptor (5) by its descriptor (family) The ith element of this array specifies the type of specifications[i].
[in] specifications	CapeArrayString	one or more specifications the compounds must match. It may contain wildcards, but the PPDB has the right, to restrict the use of wildcards to certain types: ? exactly one character * a string of zero or more characters The minimum functionality a PPDB must have is: name wildcards allowed, name may begin with a wildcard formula Sum formula of the compounds looked for. Upper and lower case letters must be used: "CLH" is illegal, "HCl" and "ClH" are

		OK. Wildcards are allowed, but the formula must not begin with a wildcard CAS number Wildcards do not make sense. DbSpecificId Wildcards do not make sense. Structure It depends on the PPDB, if wildcards are allowed Descriptors It depends on then PPDB, if wildcards are allowed
[out] numberOfSubstances	CapeShort	number of returned substances
[out] substances	CapeArrayPpdbCompound	A list of compounds that match the given specifications with "numberOfSubstances" elements

Errors

ECapeNoImpl: method not implemented

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapePersistenceNotFound: Table of compounds not available.

ECapeBadArgument: Unknown specification given, compound cannot be found

Interface Name	ICapePpdb
Method Name	GetStructure
Returns	--

Description

Delivers the chemical two-dimensional structure of a given compound. This method needs not to be implemented. In this case, the argument isImplemented[7] of "open" has to be false.

Arguments

Name	Type	Description
[in] specType	CapeShort	Denotes how the compound is specified in the next argument as symbolic constant SPEC... CAPE_SpecCompoundFormula (0) by its sum formula CAPE_SpecCompoundName (1) by its name (synonym or systematic or internal name) CAPE_SpecCompoundCasNo (2) by its CAS number CAPE_SpecCompoundDbSpecificId (3) by its data base specific ID CAPE_SpecCompoundStructure (4) by its structure CAPE_SpecCompoundDescriptor (5) by its descriptor (family)
[in] specification	CapeString	Specification which defines a substance unambiguously. Wildcards should not be used.
[in,out] structureFormat	CapeShort	Format in which the chemical structure should be and is actually delivered. Cape_SpecStructureDDB (0): Artist format[2] Cape_SpecStructureMDL (1): MDL format[3] Cape_SpecStructureSMILES (2): SMILES format[4]
[out] chemicalStructure	CapeString	The data base may select another format for transferring the chemical structure

Errors

ECapePersistenceNotFound: Structure for the given compound not found

ECapeNoImpl: Method is not implemented

ECapeBadArgument: Specified compound not present in data base

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

Interface Name	ICapePpdb
Method Name	GetCompounds
Returns	--

Description

This method gets the compounds of the mixture that belongs to a data table. It can only be called after a query was done with the aid of the methods "QueryForTables", "ExtendedQueryForTables", "QueryForModels", "ExtendedQueryForModels" or "QueryTableID". If this condition is violated, the method will store FALSE in the argument "isOK".

Arguments

Name	Type	Description
[in] rankNo	CapeShort	Rank number, obtained from "GetTableDescription" or "GetModelDescription". 0: Read compound or mixture information on the same table that was read with the last call of "GetTableData" or "GetModelParameters". A sequential read is not supported.
[out] numberOfCompounds	CapeShort	number of compounds of the chemical system 1, if the chemical system is a pure compound Should be CapeArrayStringUNDEFINED, if request is rejected
[out] compounds	CapeArrayPpdbCompound	the compounds of the chemical system (either a pure compound or a mixture) cardinality: "numberOfCompounds" This array of C-structures has a very complicated form in DCOM: VARIANT SAFEARRAY of VARIANTs (for each compound) VARIANT of CapeString VARIANT of CapeString VARIANT of CapeString VARIANT of CapeString VARIANT of CapeString VARIANT of (VARIANT of SAFEARRAY of CapeString) VARIANT of (VARIANT of SAFEARRAY of CapeString)
[out] purities	CapeArrayString	their purities as text with unit ith element = CapeStringUNDEFINED: purity of ith compound is not available cardinality: "numberOfCompounds" Should be CapeArrayStringUNDEFINED, if request is rejected
[out] reactionStoichiometry	CapeArrayShort	the stoichiometric number of each compound (educts are negative, products positive) e.g.

		2 C2H6 + 7 O2 = 4 CO2 + 6 H2O compounds = [C2H6, O2, CO2, H2O] reactionStoichiometry = [-2, -7, 4, 6] cardinality: "numberOfCompounds" Should be CapeArrayShortUNDEFINED, if request is rejected or if there is no chemical reaction
[out] isOK	CapeBoolean	TRUE: read OK FALSE: requested rankNo not found or rankNo=0 specified without foregoing "GetTableData" or "GetModelParameters".

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

Interface Name	ICapePpdb
Method Name	GetBibliography
Returns	--

Description

This method gets the bibliographic specifications of the paper or the book, in which a data table was published. It can only be called after a query was done with the aid of the methods "QueryForTables", "ExtendedQueryForTables", "QueryForModels", "ExtendedQueryForModels" or "QueryTableID". If this condition is violated, the method will store FALSE in the argument "isOK".

Arguments

Name	Type	Description
[in] rankNo	CapeShort	Rank number obtained from "GetTableDescription" or "GetModelDescription". 0: Read bibliographic information on the same table that was read with the last call of "GetTableData" or "GetModelParameters". A sequential read is not supported.
[out] referenceID	CapeString	a unique (within the database) alphanumeric identification of the publication (for citing) Should be CapeStringUNDEFINED, if request is rejected
[out] authors	CapeString	authors of the publication (family name, initials) If there is more than one author, they are separated from each other by a semicolon. Should be CapeStringUNDEFINED, if request is rejected
[out] publisher	CapeString	publisher of a book, e.g. "Longman, London"
[out] journal	CapeString	Title of the journal, may be abbreviated or not (depending on implementation) Should be CapeStringUNDEFINED, if request is rejected
[out] coden	CapeString	the journal's CODEN (This is unique, unambiguous identifier for titles of serial publications managed by the Chemical Abstracts service) Should be CapeStringUNDEFINED, if request is rejected
[out] issn	CapeString	international number of serials Should be CapeStringUNDEFINED, if request is rejected or if the source is a book
[out] titleOfBook	CapeString	title of the book Should be CapeStringUNDEFINED, if request is rejected or if the source is a journal
[out] isbn	CapeString	international number of books Should be CapeStringUNDEFINED, if request is rejected or if the source is a journal
[out] edition	CapeString	edition of a book

		Should be CapeStringUNDEFINED, if request is rejected or if the source is a journal
[out] volume	CapeString	number of the volume of a journal or a book series Should be CapeStringUNDEFINED, if request is rejected
[out] issue	CapeString	number of the issue Should be CapeStringUNDEFINED, if request is rejected
[out] page	CapeString	page or pages (from-to) of the article Should be CapeStringUNDEFINED, if request is rejected
[out] year	CapeString	year of publication Should be CapeStringUNDEFINED, if request is rejected
[out] titleOfPaper	CapeString	title of the article Should be CapeStringUNDEFINED, if request is rejected
[out] isOK	CapeBoolean	TRUE: read OK FALSE: requested rankNo not found or rankNo=0 specified without foregoing "GetTableData" or "GetModelParameters".

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

3.6.4 ICapePpdbTables

Interface Name	ICapePpdbTables
Method Name	QueryForTables
Returns	--

Description

This method and "ExtendedQueryForTables" search in a PPDB for a list of data tables, which match a given set of search conditions. Model parameters are not searched for. If no search condition is given, all tables that are stored in the PPDB are delivered. This method only prepares the Query, it does neither deliver results such as "number of records found" nor any data. Subsequent calls of a method "Get..." are necessary. If the cardinality of the answer-set is too big, ECapeLimitedImpl will be returned or raised.

The only difference between this method and "ExtendedQueryForTables" is that the latter has a lot of additional arguments, which allow more complicated search inquiries.

If both arguments "chemicalSystem" and "properties" are specified, they are combined by "AND".

If this method does not find a table, it may - but needs not - raise an exception or return a code ECapePersistenceNotFound; all subsequent calls of "Get..." will return false.

Arguments

Name	Type	Description
[in] numberOfCompounds	CapeShort	If a mixture is specified: number of compounds of that mixture If a pure compound is searched for: 1 0: The number of compounds is undefined, i.e. search for all pure compounds and mixtures that match the specifications given in the argument "chemicalSystem" CapeShortUNDEFINED: No search for certain chemical substances, i.e. all pure compounds and mixtures are searched for
[in] types	CapeArrayShort	types of the specifications of the compounds (next parameter "chemicalSystem") as symbolic constant SPEC... CAPE_SpecCompoundFormula (0) by its sum formula CAPE_SpecCompoundName (1) by its name (synonym or systematic or internal name) CAPE_SpecCompoundCasNo (2) by its CAS number CAPE_SpecCompoundDbSpecificId (3) by its data base specific ID CAPE_SpecCompoundStructure (4) by its structure CAPE_SpecCompoundDescriptor (5) by its descriptor (family)

		The cardinality of this array must be equal to the cardinality of "chemicalSystem".
[in] chemicalSystems	CapeArrayString	List of specifications - the type of the i^th element is determined by type[i] - of the compounds which are part of the mixture. If a pure compound is searched for specification of the pure compound. If a mixture is searched for, there is at most one array element for each compound of the mixture. There may be compounds that are not specified (corresponding element set to CapeStringUNDEFINED). Wildcards may be used, but their use is not recommended. If several copound specifications are given, they are combined by AND. Cardinality: at most "numberOfCompounds" elements CapeArrayStringUNDEFINED: no specification of chemical compounds, search for all pure compounds or mixtures with "numberOfCompounds" - if specified - compounds.
[in] querySubmixtures	CapeBoolean	FALSE: normal search TRUE: query for submixtures: search for a mixture and for all submixtures and pure compounds of which the mixture consists. This option requires that the mixture is fully specified, i.e. "numberOfCompounds" > 0, <> CapeShortUNDEFINED, "chemicalSystem" <> CapeStringUNDEFINED and each entry in "chemicalSystem" designates a compound unambiguously, i.e. no wildcards must be used. Example: mixture water/ethanol/1-propanol/1-butanol submixture water/ethanol/1-propanol/1-butanol OR water/ethanol/1-propanol OR water/ethanol/1-butanol OR water/1-propanol/1-butanol OR ethanol/1-propanol/1-butanol OR water/ethanol OR water/1-propanol OR water/1-butanol OR ethanol/1-propanol OR ethanol/1-butanol OR 1-propanol/1-butanol OR water OR ethanol OR 1-propanol OR 1-butanol
[in] properties	CapeArrayString	a list of properties searched for according to the property list given in Appendix A If the cardinality of this list is > 1, then the properties are combined by OR. CapeArrayStringUNDEFINED: No specification of properties, search for all properties

Errors

ECapePersistenceNotFound: No table found that matches the given search conditions.

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeBadArgument: Illegal argument of method call

ECapeNoMemory: allocation error

ECapeLimitedImpl: Too many tables found; query too unspecific

ECapeNoImpl: This special query is not implemented

Interface Name	ICapePpdbTables
Method Name	ExtendedQueryForTables
Returns	--

Description

This method and "QueryForTables" search in a PPDB for a list of data tables, which match a given set of search conditions. They do not search for model parameters. If no search condition is given, all tables that are stored in the PPDB are delivered. This method only prepares the Query, it does neither deliver results such as "number of records found" nor any data, subsequent calls of a method "Get..." are necessary. If the cardinality of the answer-set is too big, ECapeLimitedImpl will be returned or raised.

If several arguments "chemicalSystem", "properties", "phaseEqInformation", "propertyPhase", phaseEqinformation", "tableInformation", "lowerTemperature", "upperTemperature", "lowerPressure", "upperPressure", "lowerMoleFraction" and "upperMoleFraction" are specified, they are combined by "AND". If several properties are given, they are combined by "OR". The contents of the field "statesOfAggregation" are related to "properties". If several chemical compounds are given, they will be combined by "AND".

If no table is found, this method may - but needs not - raise an exception or return a code ECapePersistenceNotFound; all subsequent calls of "Get..." will return false.

Arguments

Name	Type	Description
[in] numberOfCompounds	CapeShort	If a mixture is specified: number of compounds of that mixture If a pure compound is searched for: 1 0: The number of compounds is undefined, i.e. search for all pure compounds and mixtures that match the specifications given in the argument "chemicalSystem" CapeShortUNDEFINED: No search for certain chemical substances, i.e. all pure compounds and mixtures are searched for
[in] types	CapeArrayShort	types of the specifications of the compound (next parameter "chemicalSystem") as symbolic constant SPEC... CAPE_SpecCompoundFormula (0) by its sum formula CAPE_SpecCompoundName (1) by its name (synonym or systematic or internal name) CAPE_SpecCompoundCasNo (2)

		by its CAS number CAPE_SpecCompoundDbSpecificId (3) by its data base specific ID CAPE_SpecCompoundStructure (4) by its structure CAPE_SpecCompoundDescriptor (5) by its descriptor (family) The cardinality of this array must be equal to the cardinality of "chemicalSystem".
[in] chemicalSystem	CapeArrayString	List of specifications - the type of the ith element is determined by type[i] - of the compounds which are part of the mixture. If a pure compound is searched for specification of the pure compound. If a mixture is searched for, there is at most one array element for each compound of the mixture. There may be compounds that are not specified (corresponding element set to CapeStringUNDEFINED). Wildcards may be used, but their use is not recommended. Several compound specifications are combined by AND. Cardinality: at most "numberOfCompounds" elements CapeArrayStringUNDEFINED: no specification of chemical compounds, search for all pure compounds or mixtures with "numberOfCompounds" - if specified - compounds.
[in] querySubmixtures	CapeBoolean	FALSE: normal search TRUE: query for submixtures: search for a mixture and for all submixtures and pure compounds of which the mixture consists. This option requires that the mixture is fully specified, i.e. "numberOfCompounds" > 0, <> CapeShortUNDEFINED, "chemicalSystem" <> CapeStringUNDEFINED and each entry in "chemicalSystem" designates a compound unambiguously, i.e. no wildcards must be used. Example: mixture water/ethanol/1-propanol/1-butanol submixture water/ethanol/1-propanol/1-butanol OR water/ethanol/1-propanol OR water/ethanol/1-butanol OR water/1-propanol/1-butanol OR ethanol/1-propanol/1-butanol OR water/ethanol OR water/1-propanol OR water/1-butanol OR ethanol/1-propanol OR ethanol/1-butanol OR 1-propanol/1-butanol OR water OR ethanol OR 1-propanol OR 1-butanol
[in] properties	CapeArrayString	a list of properties searched for according to the property list given in Appendix A If the cardinality of this list is > 1, then the properties are combined by OR. CapeArrayStringUNDEFINED: No specification of properties, search for all properties
[in] phaseEqInformation	CapeString	type of phase equilibrium according to appendix D CapeStringUNDEFINED: phase equilibrium type is not specified
[in]	CapeArrayString	The ith element of this array denotes the state of aggregation for the

statesOfAggregation		i^{th} property according to Appendix E cardinality of the array = cardinality of "properties" CapeArrayStringUNDEFINED phase of properties are not specified. i^{th} element = CapeStringUNDEFINED: i^{th} property may belong to any phase
[in] tableInformation	CapeString	specifications of the whole data table, separated from each other by a comma, according to Appendix F CapeStringUNDEFINED: no specification
[in] addPropInformation	CapeString	A specification that applies for a property. A list of such specifications is given in Appendix G, but each data base may add its own specifications to that list. The method "ExpandDictionary" may be used for getting the recent version of that list. Several of such specifications have to be separated by commas. CapeStringUNDEFINED: no specification
[in] lowerTemperature	CapeFloat	Lower limit of a range of temperatures in K, the values should belong to. This range is open, if upperTemperature = CapeFloatUNDEFINED, i.e. search for values at temperatures > lowerTemperature. lowerTemperature and upperTemperature = CapeFloatUNDEFINED: No search for a temperature range Search for exactly one temperature: lowerTemperature = upperTemperature
[in] upperTemperature	CapeFloat	Upper limit of a range of temperatures in K, the values should belong to. This range is open, if lowerTemperature = CapeFloatUNDEFINED,i.e. search for values at temperatures < upperTemperature. lowerTemperature and upperTemperature = CapeFloatUNDEFINED: No search for a temperature range Search for exactly one temperature: lowerTemperature = upperTemperature
[in] lowerPressure	CapeFloat	Lower limit of a range of pressures in bar, the values should belong to. This range is open, if lowerPressure = CapeFloatUNDEFINED,i.e. search for values at pressures < upperPressure. lowerPressure and upperPressure = CapeFloatUNDEFINED: No search for a pressure range Search for exactly one pressure: lowerPressure = upperPressure
[in] upperPressure	CapeFloat	Lower limit of a range of pressures in bar, the values should belong to. This range is open, if upperPressure = CapeFloatUNDEFINED,i.e. search for values at pressures > lowerPressure. lowerPressure and upperPressure = CapeFloatUNDEFINED: No search for a pressure range Search for exactly one pressure: lowerPressure = upperPressure
[in]lowerMoleFraction	CapeFloat	Lower limit of a range of compositions, the values should belong to. This range is open, if upperMoleFraction = CapeFloatUNDEFINED, i.e. search for values at compositions > lowerMoleFraction. lowerMoleFraction and upperMoleFraction =

		CapeFloatUNDEFINED: No search for a composition range Search for exactly one composition: lowerMoleFraction = upperMoleFraction *Note: Compositions can only be specified in mole fractions. The data base is not expected to convert other composition units into mole fractions during the search.*
[in] upperMoleFraction	CapeFloat	Upper limit of a range of compositions, the values should belong to. This range is open, if lowerMoleFraction = CapeFloatUNDEFINED, i.e. search for values at compositions < upperMoleFraction. lowerMoleFraction and upperMoleFraction = CapeFloatUNDEFINED: No search for a composition range Search for exactly one composition: lowerMoleFraction = upperMoleFraction *Note: Compositions can only be specified in mole fractions. The data base is not expected to convert other composition units into mole fractions during the search.*
[in] rankNoOfChemicalSystem	CapeShort	Specifies the compound for which the range of compositions ("lowerMoleFraction", "upperMoleFraction") as rank number of the argument "chemical system". Must be equal or smaller than the cardinality of "chemicalSystem".

Errors

ECapePersistenceNotFound: Table not found, query was unsuccessful.

ECapeInvalidOperation: Database is not open.

ECapeBadArgument: Illegal argument of method call

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapeLimitedImpl: Too many tables found; query too unspecific

ECapeNoImpl: This special query is not implemented

Interface Name	ICapePpdbTables
Method Name	GetTableDescription
Returns	--

Description

This method is used after a call of "QueryForTables" or "ExtendedQueryForTables" and delivers the description of the tables. These tables are processed sequentially. In order to optimize the number of calls to this method, more than one table can be transferred with one method call. These data tables are managed by rank numbers. A rank number is a sequence number from 1 to the number of tables found by the last query. A list of rank numbers is kept by the data base server. These rank numbers will be used by the other "Get..." methods.

After all tables found have been worked off, the argument "isOK" is set to FALSE. If this occurs the first time, a request may have been partially fulfilled, i.e. less tables than requested have been transferred.

If all retrieved data shall be fetched, this method has to be called as long as the argument "isOK" is TRUE.

This method should only be used to get an overview of the data tables found.

Arguments

Name	Type	Description
[in] numberOfTablesWanted	CapeShort	The number of tables that shall be read with one method call. The data base has the right to deliver fewer tables than requested. CapeShortUNDEFINED: The data base may decide how many tables will be transferred with one method call.
[out] firstRankNo	CapeShort	A rank number is a sequence number that counts the delivered table descriptions (1, 2, 3, ..., n) It may be used as input to the other "Get..." methods. "firstRankNo" contains the rank number of the first transferred table, "lastRankNo" the one of the last transferred table. The number of transferred tables is "lastRankNo" - "firstRankNo" + 1. "firstRankNo" = CapeShortUNDEFINED means, request is rejected and no table information is transferred.
[out] lastRankNo	CapeShort	See above
[out] numbersOfCompounds	CapeArrayShort	The number of compounds of the mixtures that belong to each data table 1: pure compound cardinality = number of transferred data tables Should be CapeArrayShortUNDEFINED, if request is rejected.
[out] compoundNames	CapeArrayString	List of names of the compounds which are part of the mixture or unique name of the pure compound. Such a list is transferred for each table, the cardinality of this array is the sum of all elements of "numbersOfCompounds" The position of the list of compounds belonging to the i^{th} data table may be obtained by summing up numbersOfCompounds[1]

		to numbersOfCompounds[i-1]. Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] tableDescriptions	CapeArrayString	For each table a short description of the properties, which are stored in the table The form of this description is not standardized. cardinality = number of transferred data tables Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] numbersOfProperties	CapeArrayShort	number of table columns for each data table cardinality = number of transferred data tables Should be CapeArrayShortUNDEFINED, if request is rejected.
[out] numbersOfDataPoints	CapeArrayShort	number of table rows for each data table cardinality = number of transferred data tables Should be CapeArrayShortUNDEFINED, if request is rejected.
[out] temperatureRanges	CapeArrayFloat	For each data table lower and upper limit of temperatures cardinality = 2 * number of transferred data tables Should be CapeArrayFloatUNDEFINED, if request is rejected.
[out] temperatureUnits	CapeArrayString	The unit of the temperature in each data table ith element = CapeStringUNDEFINED: ith table has no temperature cardinality = number of transferred data tables Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] pressureRanges	CapeArrayFloat	For each data table lower and upper limit of pressures cardinality = 2* number of transferred data tables Should be CapeArrayFloatUNDEFINED, if request is rejected.
[out] pressureUnits	CapeArrayString	The unit of the pressure in each data table ith element = CapeStringUNDEFINED: ith table has no pressure cardinality = number of transferred data tables Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] isOK	CapeBoolean	TRUE: Request is completely fulfilled FALSE: No more tables found, request may be only partially fulfilled.

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapeBadArgument: Argument "numberOfTablesWanted" must be ≥ 0.

Interface Name	ICapePpdbTables
Method Name	GetTableData
Returns	--

Description

This method reads a whole data table that has been previously retrieved by "QueryForTables", "ExtendedQueryForTables" or "QueryTableID" from a PPDB. This reading may be done either sequentially or directly by using the rank number obtained from a previous call of "GetTableDescription". A sequential read may be repeated until the method returns FALSE in the argument "isOK". The method may return FALSE, when the last table is transferred, but it must reject the call and return FALSE, if the last table has already been sent. Additional information on the properties and values can be obtained by "**GetTableDataInformation**".

Arguments

Name	Type	Description
[in] rankNo	CapeShort	either a rank number obtained from "GetTableDescription" or 0 (sequential read, beginning with the actual position)
[in] siWanted	CapeBoolean	TRUE: If possible, the values shall be delivered in standard SI-units FALSE: The values shall be delivered as they are stored in the data bank *Note: There are data banks which cannot convert values to SI-units*
[out] numberOfProperties	CapeShort	number of table columns CapeShortUNDEFINED: Request is rejected, no more data available
[out] properties	CapeArrayString	names of the properties that are present in the table columns according to Appendix A, e.g. [0] "temperature" [1] "pressure" [2] "heatCapacity" cardinality of array: "numberOfProperties" Should be CapeArrayStringUNDEFINED, if request is rejected
[out] relatedToCompoundNo	CapeArrayShort	For each partial molar property there is a number between 1 and the number of compounds in the mixture that specifies, which compound the property is related to. If the ith property is not a partial molar one, then the ith element of this array should be CapeShortUNDEFINED. CapeArrayShortUNDEFINED for pure compounds. cardinality of array: "numberOfProperties" Should be CapeArrayShortUNDEFINED, if request is rejected
[out] phaseEqInformation	CapeString	Type of phase equilibrium according to Appendix D. Should be CapeStringUNDEFINED, if request is rejected

[out] tableInformation	CapeString	additional information on the whole data table, may consist of one or several items from Appendix F or one or several free texts. These items have to be separated from each other by commas. Should be CapeStringUNDEFINED, if request is rejected
[out] statesOfAggregation	CapeArrayString	The ith element of this array denotes the state of aggregation that belongs to the ith property according to Appendix E cardinality of array: "numberOfProperties" Should be CapeArrayStringUNDEFINED, if request is rejected
[out] numberOfDataPoints	CapeShort	number of table rows Should be CapeShortUNDEFINED, if request is rejected
[out] values	CapeArrayFloat	numerical values of all columns and all rows of the data table. Let a table have m columns and n rows (numberOfProperties=m, numberOfDataPoints=n) [0] ... [m-1] 1st row [m] ... [2m-1] 2nd row [(n-1)*m] ... [n*m-1] nth row Missing values get the value CapeFloatUNDEFINED Should be CapeArrayFloatUNDEFINED, if request is rejected.
[out] units	CapeArrayString	units of the data according to Appendix B, e.g. [0] "K" [1] "Pa" [2] "J/mol.K" cardinality of array: "numberOfProperties" Should be CapeArrayStringUNDEFINED, if request is rejected
[return] isOK	CapeBoolean	TRUE: read OK FALSE: EOT encountered

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapeBadArgument: Argument "rankNo" must be ≥ 0.

Interface Name	ICapePpdbTables
Method Name	GetTableDataInformation
Returns	--

Description

This method gets the experimental errors and other information of a data table, that has been previously retrieved by a "Query..." method. It should be called after having invoked "GetTableData".

Arguments

Name	Type	Description
[in] rankNo	CapeShort	The rank number obtained from "GetTableDescription" 0: Read additional information on the same table that was read with the last call of "GetTableData". A sequential read is not supported.
[in] siWanted	CapeBoolean	TRUE: If possible, the experimental errors shall be delivered in standard SI-units This is only possible with absolute errors. FALSE: The errors shall be delivered as they are stored in the data bank *Note: There are data banks which cannot convert values to SI-units*
[out] expErrors	CapeArrayFloat	experimental errors of all columns and all rows. = CapeFloatUNDEFINED: No error given for that column and row Let a table have m columns and n rows (obtained by "GetTableData"): [0] ... [m-1] 1st row [m] ... [2m-1] 2nd row [(n-1)*m] ... [n*m-1] nth row cardinality of array: "numberOfProperties" * "numberOfDataPoints" (m*n, see method "GetTableData") Should be CapeArrayFloatUNDEFINED, if request is rejected
[out] errorUnits	CapeArrayString	units of the errors according to Appendix B. Relative errors have no unit, errors given as percentage have the unit "%". Example: [0] CapeStringUNDEFINED [1] CapeStringUNDEFINED [2] "J/mol.K" Should be CapeArrayStringUNDEFINED, if request is rejected
[out] addPropInformation	CapeArrayString	additional specifications of the properties for each table column, according to Appendix G The ith specification belongs to the ith property.

		If there are several specifications for a single data column, they are separated from each other by a comma. Cardinality of array: "numberOfProperties" (see method "GetTableData") Should be CapeArrayStringUNDEFINED, if request is rejected
[out] setInformation	CapeArrayString	Additional specifications for each data point (table row), according to Appendix G If the are several specifications for a single data point, they are separated from each other by a comma. The jth specification belongs to the jth data point. if the jth specification is CapeStringUNDEFINED, then there is no setInformation for the jth data point. cardinality of array: "numberOfDataPoints" (see method "GetTableData") Should be CapeArrayStringUNDEFINED, if request is rejected or if no data point has an additional specification.
[out] notes	CapeArrayString	Comments for each data value (each table column and each table row) If there are several comments for a single value, they should be separated from each other by a comma. Let a table have m columns and n rows (obtained by "GetTableData"): [0] ... [m-1] 1st row [m] ... [2m-1] 2nd row [(n-1)*m] ... [n*m-1] nth row cardinality of array: "numberOfProperties" * "numberOfDataPoints" (see method "GetTableData") Should be CapeArrayStringUNDEFINED, if request is rejected or if none of the values has a comment.
[out] isOK	CapeBoolean	TRUE: read OK FALSE: A table with the given rank number is not available

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapeBadArgument: Argument "rankNo" must be ≥ 0

3.6.5 ICapePpdbModels

Interface Name	ICapePpdbModels
Method Name	QueryForModels
Returns	--

Description

This method and "ExtendedQueryForModels" search in a PPDB for a list of model parameter sets, which match a given set of search conditions. They do not search for tables with property values. If no search condition is given, all parameter sets that are stored in the PPDB are delivered. This method only prepares the query, it does neither deliver results such as "number of records found" nor any data, subsequent calls of a method "Get..." are necessary. If the cardinality of the answer-set is too big, ECapeLimitedImpl will be returned or raised.

If several of the arguments "chemicalSystem", "modelName" and "property" are specified, they are combined by "AND".

If this method does not find a set of model parameters, it may - but needs not - raise an exception or return a code ECapePersistenceNotFound; all subsequent calls of "Get..." will return false.

Arguments

Name	Type	Description
[in] numberOfCompounds	CapeShort	If a mixture is specified: number of compounds of that mixture If a pure compound is searched for: 1 0: The number of compounds is undefined, i.e. search for all pure compounds and mixtures that match the specifications given in the argument "chemicalSystem" CapeShortUNDEFINED: No search for certain chemical substances, i.e. all pure compounds and mixtures are searched for
[in] types	CapeArrayShort	types of the specifications of the compound (next parameter "chemicalSystem") as symbolic constant SPEC... CAPE_SpecCompoundFormula (0) by its sum formula CAPE_SpecCompoundName (1) by its name (synonym or systematic or internal name) CAPE_SpecCompoundCasNo (2) by its CAS number CAPE_SpecCompoundDbSpecificId (3) by its data base specific ID CAPE_SpecCompoundStructure (4) by its structure CAPE_SpecCompoundDescriptor (5) by its descriptor (family) The cardinality of this array must be equal to the cardinality of

		"chemicalSystem".
[in] chemicalSystem	CapeArrayString	List of specifications - the type of the ith element is determined by type[i] - of the compounds which are part of the mixture. If a pure compound is searched for specification of the pure compound. If a mixture is searched for, there is at most one array element for each compound of the mixture. There may be compounds that are not specified (corresponding element set to CapeStringUNDEFINED). Wildcards may be used, but their use is not recommended. Cardinality: at most "numberOfCompounds" elements CapeArrayStringUNDEFINED: no specification of chemical compounds, search for all pure compounds or mixtures with "numberOfCompounds" - if specified - compounds.
[in] querySubmixtures	CapeBoolean	FALSE: normal search TRUE: query for submixtures: search for a mixture and for all submixtures and pure compounds of which the mixture consists. The search terms are combined by "OR". This option requires that the mixture is fully specified, i.e. "numberOfCompounds" > 0, <> CapeShortUNDEFINED, "chemicalSystem" <> CapeStringUNDEFINED and each entry in "chemicalSystem" designates a compound unambiguously, i.e. wildcards must not be used. Example: mixture water/ethanol/1-propanol/1-butanol submixture water/ethanol/1-propanol/1-butanol OR water/ethanol/1-propanol OR water/ethanol/1-butanol OR water/1-propanol/1-butanol OR ethanol/1-propanol/1-butanol OR water/ethanol OR water/1-propanol OR water/1-butanol OR ethanol/1-propanol OR ethanol/1-butanol OR 1-propanol/1-butanol OR water OR ethanol OR 1-propanol OR 1-butanol
[in] modelName	CapeString	This parameter allows to search for a certain model equation. The name of a model must be taken from appendix C. For some model names, e.g. "polynomial", it does make sense to search for a property as well. CapeStringUNDEFINED: No specification of a model, search for all data
[in] property	CapeString	A property calculated by the model (dependent) as listed in Appendix A CapeStringUNDEFINED: No specification of a property, search for all properties

Errors

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapeBadArgument: An argument of the method call is incorrect.

ECapeLimitedImpl: Too many parameter sets found; query too unspecific

ECapeNoImpl: This special query is not implemented

ECapeInvalidOperation: Database is not open.

ECapePersistenceNotFoud: No set of model parameters matches the query

Interface Name	ICapePpdbModels
Method Name	ExtendedQueryForModels
Returns	--

Description

This method and "QueryForModels" search in a PPDB for a list of model parameter sets, which match a given set of search conditions. They do not search for tables with property values. If no search condition is given, all parameter sets that are stored in the PPDB are delivered. This method only prepares the query, it does neither deliver results such as "number of records found" nor any data, subsequent calls of a method "Get..." are necessary. If the cardinality of the answer-set is too big, ECapeLimitedImpl will be returned or raised.

If several of the arguments "chemicalSystem", "modelName", "property", "stateOfAggregation", modelSpecificInformation", "lowerTemperature", "upperTemperature", "lowerPressure", "upperPressure", "lowerMoleFraction" and "upperMoleFraction" are specified, they are combined by "AND".

If this method does not find a set of model parameters, it may - but needs not - raise an exception or return a code ECapePersistenceNotFound; all subsequent calls of "Get..." will return false.

Arguments

Name	Type	Description
[in] numberOfCompounds	CapeShort	If a mixture is specified: number of compounds of that mixture If a pure compound is searched for: 1 0: The number of compounds is undefined, i.e. search for all pure compounds and mixtures that match the specifications given in the argument "chemicalSystem" CapeShortUNDEFINED: No search for certain chemical substances, i.e. all pure compounds and mixtures are searched for
[in] types	CapeArrayShort	types of the specifications of the compound (next parameter "chemicalSystem") as symbolic constant SPEC... CAPE_SpecCompoundFormula (0) by its sum formula CAPE_SpecCompoundName (1) by its name (synonym or systematic or internal name) CAPE_SpecCompoundCasNo (2) by its CAS number

		CAPE_SpecCompoundDbSpecificId (3) by its data base specific ID CAPE_SpecCompoundStructure (4) by its structure CAPE_SpecCompoundDescriptor (5) by its descriptor (family) The cardinality of this array must be equal to the cardinality of "chemicalSystem".
[in] chemicalSystem	CapeArrayString	List of specifications - the type of the ith element is determined by type[i] - of the compounds which are part of the mixture. If a pure compound is searched for specification of the pure compound. If a mixture is searched for, there is at most one array element for each compound of the mixture. There may be compounds that are not specified (corresponding element set to CapeStringUNDEFINED). Wildcards may be used, but their use is not recommended. Cardinality: at most "numberOfCompounds" elements CapeArrayStringUNDEFINED: no specification of chemical compounds, search for all pure compounds or mixtures with "numberOfCompounds" - if specified - compounds.
[in] querySubmixtures	CapeBoolean	FALSE: normal search TRUE: query for submixtures: mixture and for all submixtures and pure compounds of which the mixture consists. The search terms are combined by "OR". This option requires that the mixture is fully specified, i.e. "numberOfCompounds" > 0, <> CapeShortUNDEFINED, "chemicalSystem" <> CapeStringUNDEFINED and each entry in "chemicalSystem" designates a compound unambiguously, i.e. no wildcards must be used. Example: mixture water/ethanol/1-propanol/1-butanol submixture water/ethanol/1-propanol/1-butanol OR water/ethanol/1-propanol OR water/ethanol/1-butanol OR water/1-propanol/1-butanol OR ethanol/1-propanol/1-butanol OR water/ethanol OR water/1-propanol OR water/1-butanol OR ethanol/1-propanol OR ethanol/1-butanol OR 1-propanol/1-butanol OR water OR ethanol OR 1-propanol OR 1-butanol
[in] modelName	CapeString	This parameter allows to search for a certain model equation. The name of a model must be taken from appendix C. For some model names, e.g. "polynomial", it does make sense to search for a property as well. CapeStringUNDEFINED: No specification of a model, search for all data
[in] property	CapeString	A property searched for according to the property list given in Appendix A CapeStringUNDEFINED: No specification of a property, search for all properties
[in]	CapeString	State of aggregation for the specified property according to

stateOfAggregation		Appendix E CapeStringUNDEFINED: Property may belong to every state of aggregation.
[in] modelSpecificInformation	CapeString	A specification that applies for a set of model parameters. A list of such specifications is given in Appendix G, but each data base may add its own specifications to that list. The method "ExpandDictionary" may be used for getting the recent version of that list. Several of such specifications have to be separated by commas. CapeStringUNDEFINED: no specification
[in] lowerTemperature	CapeFloat	Lower limit of a range of temperatures in K, the values or model parameters should belong to. This range is open, if upperTemperature = CapeFloatUNDEFINED, i.e. search for values at temperatures > lowerTemperature. lowerTemperature and upperTemperature = CapeFloatUNDEFINED: No search for a temperature range Search for exactly one temperature: lowerTemperature = upperTemperature
[in] upperTemperature	CapeFloat	Upper limit of a range of temperatures in K, the values or model parameters should belong to. This range is open, if lowerTemperature = CapeFloatUNDEFINED,i.e. search for values at temperatures < upperTemperature. lowerTemperature and upperTemperature = CapeFloatUNDEFINED: No search for a temperature range Search for exactly one temperature: lowerTemperature = upperTemperature
[in] lowerPressure	CapeFloat	Lower limit of a range of pressures in bar, the values or model parameters should belong to. This range is open, if upperPressure = CapeFloatUNDEFINED,i.e. search for values at pressures > lowerPressure. lowerPressure and upperPressure = CapeFloatUNDEFINED: No search for a pressure range Search for exactly one pressure: lowerPressure = upperPressure
[in] upperPressure	CapeFloat	Lower limit of a range of pressures in bar, the values or model parameters should belong to. This range is open, if upperPressure = CapeFloatUNDEFINED, i.e. search for values at pressures > lowerPressure. lowerPressure and upperPressure = CapeFloatUNDEFINED: No search for a pressure range Search for exactly one pressure: lowerPressure = upperPressure

Errors

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapeBadArgument: An argument of the method call is incorrect.

ECapeLimitedImpl: Too many parameter sets found; query too unspecific

ECapeNoImpl: This special query is not implemented

ECapeInvalidOperation: Database is not open.

ECapePersistenceNotFoud: No set of model parameters matches the query

Interface Name	ICapePpdbModels
Method Name	QueryTableID
Returns	--

Description

This method is used to obtain a table the ID of which is known. It only prepares the query, it does neither deliver results such as "number of records found" nor any data, subsequent calls of a method "Get..." are necessary. *The only practical use of this method is to retrieve a table with original data that are used to calculate model parameters (see method "GetModelParametersInformation").*

If this method fails, the server is in the same state as if no query has been executed. That means, all subsequent calls of "Get..." will return false.

Arguments

Name	Type	Description
[in] tableID	CapeString	ID of the wanted table A table ID may consist of any combination of alphanumeric characters, provided it is unique throughout a PPDB. The only way for the client to procure a table ID is via the argument "tableIdsOriginalData" of the method "GetModelParametersInformation". This restriction does make sense, since this method should only be used to Get access to the original data of a set of model parameters.

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

ECapePersistenceNotFound: Unknown tableID

Interface Name	ICapePpdbModels
Method Name	GetModelDescription
Returns	--

Description

This method is used after a call of "QueryForModels" or "ExtendedQueryForModels" and delivers a description of the found sets of model parameters. These sets are processed sequentially. In order to optimize the number of calls to this method, more than one set can be transferred with one method call.

The model parameter sets are managed by rank numbers. A rank number is a sequence number from 1 to the number of sets found by the last query. A list of rank numbers is kept by the data base server. These rank numbers will be used by the other "Get..." methods.

After all sets found have been worked off, the argument "isOK" is set to FALSE. If this occurs the first time, a request may have been partially fulfilled, i.e. less sets than requested have been transferred.

If all retrieved data shall be fetched, this method has to be called as long as the argument "isOK" is TRUE.

This method should only be used in order to get an overview of the model parameter sets found.

Arguments

Name	Type	Description
[in] numberOfSetsWanted	CapeShort	The number of parameter sets that shall be read with one method call. The data base has the right to deliver fewer sets than requested. CapeShortUNDEFINED: The data base may decide how many sets will be transferred with one method call.
[out] firstRankNo	CapeShort	A rank number is a sequence number that counts the delivered set descriptions (1, 2, 3, ...). It may be used as input to the other "Get..." methods. "firstRankNo" contains the rank number of the first transferred set, "lastRankNo" the one of the last transferred set. The number of transferred sets is "lastRankNo" - "firstRankNo" + 1. "firstRankNo" = CapeShortUNDEFINED means, request is rejected and no set information is transferred.
[out] lastRankNo	CapeShort	See above
[out] numbersOfCompounds	CapeArrayShort	The number of compounds of the mixtures that belong to each parameter set 1: pure compound cardinality = number of transferred parameter sets Should be CapeArrayShortUNDEFINED, if request is rejected.
[out] compoundNames	CapeArrayString	List of names of the compounds which are part of the mixture or unique name of the pure compound. Such a list is transferred for each set, the cardinality of this array is the sum of all elements of "numbersOfCompounds"

		The position of the list of compounds belonging to the ith parameter set may be obtained by summing up numbersOfCompounds[1] to numbersOfCompounds[i-1]. Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] modelNames	CapeArrayString	Names of the models that belong to the transferred parameter sets according to Appendix C. cardinality = number of transferred parameter sets Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] numbersOfIndependent Properties	CapeArrayShort	number of independent properties for each set An independent property is located on the right hand side of the model equation. In most cases it will be a temperature, a pressure or a concentration cardinality = number of transferred parameter sets Should be CapeArrayShortUNDEFINED, if request is rejected.
[out] limitsOfValidity	CapeArrayFloat	For each parameter set lower and upper limit of validity of the independent properties cardinality = 2 * number of transferred parameter sets * ∑numbersOfIndependentProperties Should be CapeArrayFloatUNDEFINED, if request is rejected.
[out] unitsForLimitsOfValidi ty	CapeArrayString	The units of the independent properties in each parameter set ith element = CapeStringUNDEFINED: ith set has no limits of validity cardinality = number of transferred parameter sets * ∑numbersOfIndependentProperties Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] isOK	CapeBoolean	TRUE: Request is completely fulfilled FALSE: No more parameter sets found, request may be only partially fulfilled.

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

Interface Name	ICapePpdbModels
Method Name	GetModel
Returns	--

Description

This method reads a model as an instance of a special model class for a parameter set that has been previously retrieved by "QueryForModels" or "ExtendedQueryForModels" from a PPDB. This reading may be done either sequentially or directly by using the rank number obtained from a previous call of "GetModelDescription". A sequential read may be repeated until the method returns FALSE in the argument "isOK".

On successful return, the server has created an instance of the interface "ICapePpdbAbstractModel" that represents the requested model. The interface must implement the methods listed in Appendix H. The instance contains all parameters and model descriptions that are needed to perform a calculation.

Servers may not implement this method.

Arguments

Name	Type	Description
[in] rankNo	CapeShort	either the rank number obtained from "GetModelDescription" or 0 (sequential read, beginning with the actual position)
[out] model	CapeInterface	Pointer to the interface "ICapePpdbAbstractModel" that implements the model, which is contained in the parameter set.
[out] isOK	CapeBoolean	TRUE: Reading was successful. FALSE: All parameter sets have been worked off.

Errors

ECapePersistenceNotFound: Class not found for specified model equation.

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

Interface Name	ICapePpdbModels
Method Name	GetModelParameters
Returns	--

Description

This method reads a set of model parameters that has been previously retrieved by "QueryForModels" or "ExtendedQueryForModels" from a PPDB. This reading may be done either sequentially or directly by using the rank number obtained from a previous call of "GetModelDescription". A sequential read may be repeated until the method returns FALSE in the argument "isOK".

Only one set of parameters is returned, even if multiple parameter sets are stored in one data table. The additional parameter sets can be obtained either by sequential read or by repeating the method call with an increased rank number.

Two types of parameter sets can be treated by this method:

(i) complete parameter sets, which are sufficient for calculating properties with the aid of an equation; e.g. paramters A, B and C of the Antoine equation

(ii) parameters which are needed by equations for calculating properties and wich are related to one or more compounds of a mixture; e.g. the binary interaction parameters of the NRTL equation.

Arguments

Name	Type	Description
[in] rankNo	CapeShort	either the rankNo obtained from "GetModelDescription" or 0 (sequential read, beginning with the actual position)
[out] modelName	CapeString	Name of the model or equation the parameters have been made for. A list of all valid models/ equations is given in Appendix C. Should be CapeStringUNDEFINED, if request is rejected
[out] dependentProperty	CapeString	Name of the property that can be calculated by the parameters according to the property list given in Appendix A, e.g. "heatCapacity" Should be CapeStringUNDEFINED, if request is rejected
[out] numberOfIndependent Properties	CapeShort	number of the independent properties (see below) If parameterType>0, 0. Should be CapeShortUNDEFINED, if request is rejected
[out] independentProperties	CapeArrayString	Names of the properties that are the independent variables according to the property list given in Appendix A, e.g. [0] "temperature" [1] "pressure" Should be CapeArrayStringUNDEFINED, if request is rejected
[out]	CapeArrayShort	If parameterType=0, there is an array element for the dependent

relatedToCompoundNo		and each independent property, which is a number between 1 and the number of compounds stating which compound the property is related to, if the property is a partial molar one CapeShortUNDEFINED, otherwise The cardinality of this array is 1 + number of independent properties If parameterType>0, there are parameterTypes array elements for each parameter, which are a number between 1 and the number of compounds stating which compound(s) the parameter is related to. Should be CapeArrayShortUNDEFINED, if request is rejected or if there are no compound related properties.
[out] phaseEqInformation	CapeString	Type of phase equilibrium according to Appendix D Should be CapeStringUNDEFINED, if request is rejected
[out] setInformation	CapeString	Additional information on the parameter set, may consist of one or several items from Appendix F or one or several free texts. The items are separated from each other by commas. Should be CapeStringUNDEFINED, if request is rejected
[out] stateOfAggregation	CapeString	State of aggregation, the dependent property is belonging to according to Appendix E Should be CapeStringUNDEFINED, if request is rejected
[out] propertyUnits	CapeArrayString	units of the properties according to Appendix B (first element dependent property, followed by the independent properties), e.g. [0] "J/mol.K" [1] "K" [2] "Pa" The cardinality of this array is 1 + "numberOfIndependentProperties" Should be CapeArrayStringUNDEFINED, if request is rejected
[out] parameterNames	CapeArrayString	names of model parameters according to Appendix C Should be CapeArrayStringUNDEFINED, if request is rejected
[out] parameters	CapeArrayDouble	value for each parameter Should be CapeArrayDoubleUNDEFINED, if request is rejected
[out] parameterType	CapeShort	Type of the parameters: 0: complete parameter set of an equation 1: unary parameter (For each parameter there is one entry in "relatedToCompoundNo" indicating which compound the parameter is related to.) 2: binary interaction parameter (For each parameter there are two entries in "relatedToCompoundNo" indicating which compounds the parameter is related to) 3: ternary interaction parameter (For each parameter there are three entries in "relatedToCompoundNo" indicating which compounds the parameter is related to) ... n: n-ary interaction parameter (For each parameter there are n entries in "relatedToCompoundNo" indicating which compounds

		the parameter is related to)
[out] limitsOfValidity	CapeArrayFloat	for each independent property lower and upper limit of validity e.g. [273.15, 473.15, 101325, 506625] The units of these limits are given in the argument "propertyUnits" - beginning with the 2nd element. The cardinality of this array is twice the cardinality of the array independentProperties Should be CapeArrayFloatUNDEFINED, if request is rejected
[out] isOK	CapeBoolean	TRUE: read OK FALSE: sequential read: EOT encountered

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

Interface Name	ICapePpdbModels
Method Name	GetModelParametersInformation
Returns	--

Description

This method extends "GetModelParameters" by delivering additional information on a set of model parameters that has been previously retrieved by "Query..." from a PPDB.

Arguments

Name	Type	Description
[in] rankNo	CapeShort	rank number obtained from "GetModelDescription". 0: Read additional information on the same parameter set that was read with the last call of "GetModelParameters" A sequential read is not supported.
[out] tableIDsOriginalData	CapeArrayString	unique ID of one or more data tables, if the data, from which the model parameters are generated, are present in the data bank This ID can be used to retrieve the values by "QueryTableID". CapeArrayStringUNDEFINED, otherwise
[out] addPropInformation	CapeArrayString	additional specifications of the properties (first element dependent property, follwed by the independent properties), separated from each other by a comma, according to Appendix G, e.g. "measured" "recommended" (quality of the data) Example [0] CapeStringUNDEFINED [1] CapeStringUNDEFINED [2] "measured, recommended" The cardinality of this array is 1 + the cardinality of the array "independentProperties" (see method "GetModelParameters") Should be CapeArrayStringUNDEFINED, if request is rejected
[out] varianceCovarianceMatrix	CapeArrayFloat	variance-covariance matrix of the parameters cardinality: (number of parameters)**2 Should be CapeArrayFloatUNDEFINED, if request is rejected
[out] estimatedPrecision	CapeFloat	the estimated error of calculation for the dependent variable Should be CapeFloatUNDEFINED, if request is rejected
[out] isOK	CapeBoolean	TRUE: read OK FALSE: EOT encountered.

Errors

ECapeInvalidOperation: Database is not open.

ECapeUnknown: internal error

ECapeNoMemory: allocation error

3.6.6 ICapePpdbAbstractModel

Such a general interface shall handle the general aspects of calculation models. It must respond to the following methods:

- GetModelContents
- GetModelInformation
- performModelCalculation

The implementation of this interface is not mandatory.

Interface Name	ICapePpdbAbstractModel
Method Name	GetModelContents
Returns	--

Description

None.

Arguments

Name	Type	Description
[out] numberOfCompounds	CapeShort	The number of compounds of the mixtures that belongs to the parameter set 1: pure compound Should be CapeShortUNDEFINED, if request is rejected.
[out] compoundNames	CapeArrayString	List of names of the compounds which are part of the mixture or unique name of the pure compound. Should be CapeArrayStringUNDEFINED, if request is rejected.
[out] modelName	CapeString	Name of the model according to Appendix C. Should be CapeStringUNDEFINED, if request is rejected.
[out] dependentProperty	CapeString	Name of the property that can be calculated by the model equation, according to the property list given in Appendix A, e.g. "heatCapacity" Should be CapeStringUNDEFINED, if request is rejected
[out] numberOfIndependent Properties	CapeShort	number of independent properties for parameter set An independent property is located on the right hand side of the model equation. In most cases it will be a temperature, a pressure or a concentration Should be CapeShortUNDEFINED, if request is rejected.
[out] limitsOfValidity	CapeArrayFloat	For each independent property lower and upper limit of validity cardinality = 2 * numberOfIndependentProperties Should be CapeArrayFloatUNDEFINED, if request is rejected.
[out] unitsForLimitsOfVali dity	CapeArrayString	The units of the independent properties cardinality = numberOfIndependentProperties Should be CapeArrayStringUNDEFINED, if request is rejected.

Errors

ECapeUnknown

Interface Name	ICapePpdbAbstractModel
Method Name	GetModelInformation
Returns	--

Description

This method extends "GetModel" by delivering additional information on a specific model.

Arguments

Name	Type	Description
[out] tableIdsOriginalData	CapeArrayString	unique ID of one or more data tables, if the data, from which the model parameters are generated, are present in the data bank This ID can be used to retrieve the values by "QueryTableID". CapeArrayStringUNDEFINED, otherwise
[out] phaseEqInformation	CapeString	Type of phase equilibrium according to Appendix D Should be CapeStringUNDEFINED, if request is rejected
[out] setInformation	CapeString	Additional information on the parameter set, may consist of one or several items from Appendix F or one or several free texts. The items are separated from each other by commas. Should be CapeStringUNDEFINED, if request is rejected
[out] stateOfAggregation	CapeString	State of aggregation, the dependent property is belonging to according to Appendix E Should be CapeStringUNDEFINED, if request is rejected
[out] addPropInformation	CapeArrayString	additional specifications of the properties (first element dependent property, follwed by the independent properties), separated from each other by a comma, according to Appendix G, e.g. "measured" "recommended" (quality of the data) Example [0] CapeStringUNDEFINED [1] CapeStringUNDEFINED [2] "measured, recommended" The cardinality of this array is 1 + the cardinality of the array "independentProperties" (see method "GetModelParameters") Should be CapeArrayStringUNDEFINED, if request is rejected
[out] estimatedPrecision	CapeFloat	the estimated error of calculation for the dependent variable Should be CapeFloatUNDEFINED, if request is rejected

Errors

ECapeUnknown

Interface Name	ICapePpdbAbstractModel
Method Name	PerformModelCalculation
Returns	--

Description

This method is used after a call of "GetModel" for performing a calculation using the retrieved model.

Arguments

Name	Type	Description
[in] initalValues	CapeArrayFloat	This array contains an inital value for each independent property.
[in] finalValues	CapeArrayFloat	This array contains a final value for each independent property.
[in] stepSizes	CapeArrayFloat	This array contains a step size for each independent property..
[in] independentUnits	CapeArrayString	This array contains a unit according to Appendix B for each independent variable.
[in,out] dependentUnit	CapeString	Input: Unit in which the result is wanted. Output: Unit, in which the result (calculatedvalues) will be given.
[out] calculatedValues	CapeArrayFloat	On return, this argument contains the calculated values. Its cardinality is the product of {(finalValues[i] -initalValues[i])/stepSizes[i] +1}over all independent variables i.

Errors

ECapeUnknown

3.6.7 Necessary extensions of CAPE-OPEN lists

This standard needs several lists of

Appendix A:	properties
Appendix B:	units
Appendix C:	methods or equations
Appendix D:	phase equilibrium information
Appendix E:	state of aggregation
Appendix F:	table information
Appendix G:	additional property information

These lists are needed to control the contents of the corresponding parameters of the methods. Their contents coincide with other CAPE-OPEN standards[6,13].

3.7 Scenarios

Any scenario demonstrating use of the previously defined interfaces

4. Interface specifications

4.1 COM IDL

```
// You can get these intructions in Ppdb.idl file from CAPE-OPENv1-0-0.zip
```

4.2 CORBA IDL

```
// You can get these intructions in CAPE-OPENv1-0-0.idl within the
CAPEOPEN100::Business::PhyProp::Ppdb module
```

5. Notes on the interface specifications

For DCOM this interface definition is not sufficient. Automation-compliant arguments are either simple arguments (short, long, float, double, string) or VARIANTs. This is a structure which contains the type of the arguments and its value. This value can be either a simple variable, a SAFEARRAY or another VARIANT. SAFEARRAYs are structures containing an array of values, its number of dimensions and its upper and lower limits. Strings are passed as BSTRs, i.e. a UNICODE string with a length descriptor.

CapeArrayLong, CapeArrayShort, CapeArrayDouble, CapeArrayFloat and CapeArrayString are modeled as VARIANTs of SAFEARRAYs of long, short, double, float and BSTR values, respectively.

"SupportedFunctions" is a VARIANT with a SAFEARRAY with exactly 7 boolean elements.

The structure type "PpdbCompounds" is modeled as a VARIANT, containing a SAFEARRAY with 7 elements, which are pointers to VARIANTS for name (BSTR), sumFormula (BSTR), casNo (BSTR), dbSpecificId (BSTR), internalName (BSTR), synonyms (SAFEARRAY of BSTRs) and descriptors (SAFEARRAY of BSTRs). "SubstanceArray" is a VARIANT with a SAFEARRAY of elements of the type "PpdbCompound".

6. Prototypes implementation

The prototype implementation was done under Visual C++ (Version 5.0). It reads a PPDX neutral file[10] (server) and displays its content on the screen (client). The server is an ATL executable, the client a Win32 console application.

The client can only be started from a DOS-window.

```
cd programme\dechema\ppdb
ppdb_client                (Server is located on the same computer.)
ppdb_client computer_name  (Server is located on a remote machine.)
```

The server consists of four types of classes:

(i) Classes that implement the interfaces (CapePpdb, CapePpdbRegister) and communicates with the class Ppdx (see below)

(ii) Class Ppdx, which implements the reading in a neutral PPDX-file. It has no COM-specific variables.

(iii) An implementation of the global interface ICapeIdentification

(iv) Auxiliary classes

All these classes are C++ classes. For constructing them the general methods of object-oriented programming can be used.

If you want to have information on general errors and syntactical errors in the PPDX files, add a system-wide environment variable:

```
PPDB_SERVER_DIAG_FILE nameOfDiagnosticFile (with a full pathname)
```

This file will be created for every run and filled with diagnostic information. This environment variable should not be present in a production environment. System wide environment variables can only be set up under an administrator account.

6.1 Class CapeIdentification

(files CapeIdentification.cpp, CapeIdentification.h)

This class implements the functionality of the general CO-interface ICapeIdentification[11].

6.2 Class CapePpdbRegister

(files CapePpdbRegister.cpp, CapePpdbRegister.h)

This class implements the functionality of the interface ICapePPdbRegister.

6.3 Class CapePpdb

(files CapePpdb.cpp, CapePpdb.h)

This is the implementation of the interfaces ICapePpdb, ICapePpdbTables and ICapePpdbModels. The method "Open" creates an instance of "Ppdx" and stores its address in a private variable. All other methods transform their arguments to standard C++ types, call the appropriate method (mostly having the same name) of the instance of Ppdx and convert the results to COM-automation data-types (SAFEARRAYs, VARIANTs)

6.4 Class Ppdx

This class is a pure C++ class. It contains methods for reading from a neutral file. They resemble the methods of the class CapePpdx, but here only classical C++ variable types are used. This makes the implementation of this class independent of the type of interfaces, it can as well used with CORBA as with DCOM.

6.5 Class StringArray

(files StringArray.cpp, StringArray.h)

This class manages arrays of strings.

6.6 Class pListen

(file pListen.cpp, pListen.h, stringproc.c, stringproc.h, strtools.c strtools.h, verk_lis.c, verk_lis.h)

This class is a special development for PPDx-files. It used for storing the contents of a PPDX-file internally.

6.7 Functions for manipulating automation types

(file manautyp.cpp, manautyp.h)

There is a set of routines which help in dealing with the complex DCOM types, especially for converting BSTRs to ASCII-strings and vice versa and for manipulating string arrays.

6.8 Repository

(files repository.c, repository.h)

This implementation uses a key HKEY_LOCAL_MACHINE\SOFTWARE\CAPE\PPDB\registry of the Windows® registry as repository. The "name" entry contains the name of the PPDB, the "value" entry has two fields, which are separated from each other by "|".

1. several sub-fields, separated from each other by a colon
    - type of PPDB, name of data base server, user-id (relational data base)
    - type of PPDB, full name of file (PPDX-file)
2. short description of the contents of the data base

Example:

name	value
DETHERM	DETHERM:deimos:stoffdb|DETHERM-Stoffdatenbank
aceton-NFF	ppdx:"C:\programme\dechema\ppdb\aceton.ikc"|Viscosity of acetone
aceton-2-NFF	ppdx:"F:\tmp\aceton2.ikc"|Antoine-Konstanten von Aceton

6.9 The client

(file ppdb-client.cpp)

This is a simple console application, which displays the contents of the repository, opens the second PPDB, expands the dictionary of properties (file "properties.dic"), searches in the PPDB for the viscosity of acetone and shows the result of the query.

For display of dictionaries, a system wide environment variable has to be set up:

```
DICTIONARY_PATH     c:\programme\dechema\ppdb
```

System wide environment variables can only be set up under an administrator account.

6.10 Installation

Client and server can be installed by executing `setup_ppdb.exe.`

For installation of the client at a remote machine, the Proxy-Stub-DLL and the server must be registered by

```
RegSvr32 ppdb_proxystub.dll
ppdb_server /RegServer
```

6.11 Desinstallation

Before you can delete the programs, the server has to be uninstalled.

```
ppdb_server /UnregServer
```

Then the normal InstallShield® deinstall procedure can be carried out.

7. Specific glossary terms

An **array** is a set with a defined order of its elements. Each element has an index value attributed to it. Arrays are implemented as SAFEARRAYs (COM) or sequences (CORBA)

The **cardinality** of a set or an array is the number of elements of that set or array.

A **component** is a part of a chemical mixture.

A **compound** is either a pure chemical substance or - "pseudocompound" - a defined mixture of chemical substances like air or petroleum cuts.

A **dependent property** will be calculated by a model equation. In most cases it is located on the left hand side of such an equation.

A **dictionary** is a set of all valid values for a certain item.

An **independent property** is located on the right hand side of a model equation. In most cases it will be a temperature, a pressure or a concentration

An **interface** is

(i) a specifier for the externally-visible operations of a class, component, or other entity without specification of internal structure[15]

(ii) a collection of interfaces ("PPDB interface")

A **mixture** consists of chemical substances with a defined composition.

A **model** is an algorithm for calculating thermophysical property data from constants that are stored in the data bank.

A **(model) parameter set** is a set of parameters which are used by a model equation for calculating property values.

A **physical property data base (PPDB)** is an abstract model for all types of collections with thermophysical property data and with parameters of models for calculating thermophysical property data that have been taken from the literature, have been measured or processed in one's own laboratory or have been obtained from other sources.

A **property** is a thermophysical property like vapor pressure, heat capacity, temperature, pressure or composition.

A **table** looks like it is published in a scientific paper. Each table has a header which defines the properties and units, and a body which contains data. It has a column per property and a row per data point.

8. Bibliography

[1] Hill, J. Am. Chem. Soc. 2288), 478-494 (1900)

[2] http://www.ddbst.de/artist.htm

[3] MDL Information Systems, Inc., 14600 Catalinie Str., San Leandro CA 94577, USA, "CTfile Formats", December 1999, http://www.mdli.com/downloads/literature/ctfile.pdf

[4] http://www.daylight.com/dayhtml/smiles/smiles-etc.html, see also: D. Weininger, "SMILES 1. Introduction and Encoding Rules", J. Chem. Inf. Comput. Sci., 28, 31 (1988)

[5] J.P.Belaud, J.Bernier, M.Halloran, J.Köller, D.Piñol, P.Roux, "Error Handling Strategy: Error Common Interface", CAPE-OPEN document Version 5, Sept. 2000

[6] "Open Interface Specifications Thermodynamic and Physical Properties, Thermo Work Package", CAPE-OPEN document CO-THRM-1 Version 1.07

[7] Aspen Plus Version 10.0 Documentation, "Physical Property Methods and Models", reference manual, 1998, Chapter 3

[8] J. Gmehling, U. Onken, "Vapor-Liquid Equlibrium Data Collection - Aqueous Organic Systems", DECHEMA Chemistry Data Series, Vol. I, Part 1, Frankfurt am Main 1977

[9] Hyprotech Ltd., HYSYS manual: Simulation Basis, Appendix A- Property Methods and Calculations

[10] http://www.dechema.de/infsys/dsd/englisch/ikcppdxMain.htm (case sensitive)

[11] J.-P. Belaud, D. Piñol: "Open Interface Specification: Identification CO Service", CAPE-OPEN document CO-M&T-20, 2000

[12] www.mico.org

[13] D. Piñol, J.C. Rodriguez: "Second Update of the CAPE-OPEN Thermo Specification", CAPE-OPEN document 2CO-Thermo Spec Update.doc, 2000

[14] The Methods and Tool Group: "CAPE-OPEN Common Interfaces: Update on Types, Interfaces Naming and Empty Values", Gobal CAPE-OPEN document CO-M&T-Update, 2000

[15] Rational Software Corporation et al: "UML Notation Guide", version 1.1, available from www.rational.com/uml

8.1 Process simulation references

8.2 Computing references

8.3 General references

9. Appendices

9.1 Appendix A: Properties

This list is considered to be preliminary. A set of properties has to be defined as a standard within CAPE-OPEN and a definition of each property has to be given. This work has to be done by another CAPE-OPEN committee.

2ndInteractionVirialCoefficient

3rdVirialCoefficient

4thVirialCoefficient

5thVirialCoefficient

6thVirialCoefficient

acentricFactor

acousticCompressibility

activationEnergyOfViscousFlow

activationVolume

activity

activityCoefficient

activityCoefficientAnion

activityCoefficientCation

associationConstant

boilingTemperature

brunauerEmmettTellerSurface

bubblePointPressure

bubblePointTemperature

bunsenAbsorptionCoefficient

burningIndex

burningTime

complexPermittivityImaginaryPart

complexPermittivityRealPart

compressibility

compressibilityCoefficient

compressibilityFactor

concentrationOfMetalWithinElectrode

concentrationSolvent

concentrationSolventInGaseousPhase

criticalCompressibilityFactor

criticalDensity

criticalPressure

criticalTemperature

criticalVolume

crystalEnergy

cubicExpansionCoefficient

cumulativeDistribution

currentDensity

density

dewPointPressure

dewPointTemperature

dielectricConstant

differentialJouleThomsonCoefficient

diffusionCoefficient

diffusionVolume

dipoleMoment

dissociationConstant

dissociationDegree

distributionCoefficient

electricalResistance

electromotiveForce

energy

energyAnion

energyCation

energyFunction(E(0))

energyFunction(E(298))

energyLennardJones

energyOfActivation

energyOfCombustion

energyOfDecomposition

energyOfDissociation

energyOfFormation

energyOfFusion

energyOfHydration

energyOfHydrationAnion

energyOfHydrationCation

energyOfIonization

energyOfIonizationAnion

energyOfIonizationCation

energyOfMicellation

energyOfMixing

energyOfPhaseTransition

energyOfReaction

energyOfSolution

energyOfSolutionAnion

energyOfSolutionCation

energyOfSolvation

energyOfSolvationAnion

energyOfSolvationCation

energyOfStabilization

energyOfSublimation

enthalpy

enthalpyAnion

enthalpyCation

enthalpyFunction(H(0))

enthalpyFunction(H(298))

enthalpyFunctionOfReaction(H(0))

enthalpyFunctionOfReaction(H(298))

enthalpyOfActivation

enthalpyOfAdsorption

enthalpyOfCombustion

enthalpyOfDecomposition

enthalpyOfDilution

enthalpyOfFormation

enthalpyOfHydration

enthalpyOfHydrationAnion

enthalpyOfHydrationCation

enthalpyOfIonization

enthalpyOfIonizationAnion

enthalpyOfIonizationCation

enthalpyOfMicellation

enthalpyOfMixing

enthalpyOfPhaseTransition

enthalpyOfReaction

enthalpyOfSolution

enthalpyOfSolutionAnion

enthalpyOfSolutionCation

enthalpyOfSolvation

enthalpyOfSolvationAnion

enthalpyOfSolvationCation

enthalpyOfTransfer

entropy

entropyAnion

entropyCation

entropyFunction(H(0))

entropyOfActivation

entropyOfCombustion

entropyOfDissociation

entropyOfFormation

entropyOfFusion

entropyOfHydration

entropyOfHydrationAnion

entropyOfHydrationCation

entropyOfIonization

entropyOfIonizationAnion

entropyOfIonizationCation

entropyOfMicellation

entropyOfMixing

entropyOfPhaseTransition

entropyOfReaction

entropyOfSolution

entropyOfSolutionAnion

entropyOfSolutionCation

entropyOfSolvation

entropyOfSolvationAnion

entropyOfSolvationCation

entropyOfSublimation

entropyOfTransfer

entropyOfVaporization

equilibriumConstant

equilibriumConstantOfFormation

evaporationNumber

excessEnergy

excessEnthalpy

excessEntropy

excessGibbsEnergy

excessHeatCapacity

excessHelmholtzEnergy

excessVolume

expansivity

flow

fraction

frequency

frequencyDistribution

fugacity

fugacityCoefficient

fusionPressure

fusionTemperature

gasConcentration

gibbsEnergy

gibbsEnergyAnion

gibbsEnergyCation

gibbsEnergyFunction(H(0))

gibbsEnergyFunction(H(298))

gibbsEnergyFunctionOfReaction(H(0))

gibbsEnergyFunctionOfReaction(H(298))

gibbsEnergyOfActivation

gibbsEnergyOfAdsorption

gibbsEnergyOfCombustion

gibbsEnergyOfDissociation

gibbsEnergyOfFormation

gibbsEnergyOfFusion

gibbsEnergyOfHydration

gibbsEnergyOfHydrationAnion

gibbsEnergyOfHydrationCation

gibbsEnergyOfIonization

gibbsEnergyOfIonizationAnion

gibbsEnergyOfIonizationCation

gibbsEnergyOfMicellation

gibbsEnergyOfMixing

gibbsEnergyOfPhaseTransition

gibbsEnergyOfReaction

gibbsEnergyOfSolution

gibbsEnergyOfSolutionAnion

gibbsEnergyOfSolutionCation

gibbsEnergyOfSolvation

gibbsEnergyOfSolvationAnion

gibbsEnergyOfSolvationCation

gibbsEnergyOfSublimation

gibbsEnergyOfTransfer

gibbsEnergyOfVaporization

gibbsSurfaceEnergy

glassTransitionTemperature

grossCalorificValue

gyrationRadius

halflifeTime

hansenDispersiveSolubilityParameter

hansenHydrogenBondingSolubilityParameter

hansenPolarSolubilityParameter

heatCapacity

heatCapacityRatio

heatOfFusion

heatOfFusionAtNormalFreezingPoint

heatOfSolidSolidPhaseTransition

heatOfSublimation

heatOfVaporization

heatOfVaporizationAtNormalBoilingPoint

heatProductionRate

heatTransferCoefficient

helmholtzEnergy

helmholtzEnergyAnion

helmholtzEnergyCation

helmholtzEnergyFunction(E(0))

helmholtzEnergyFunction(E(298))

helmholtzEnergyOfActivation

helmholtzEnergyOfDissociation

helmholtzEnergyOfFormation

helmholtzEnergyOfFusion

helmholtzEnergyOfHydration

helmholtzEnergyOfHydrationAnion

helmholtzEnergyOfHydrationCation

helmholtzEnergyOfIonization

helmholtzEnergyOfIonizationAnion

helmholtzEnergyOfIonizationCation

helmholtzEnergyOfMicellation

helmholtzEnergyOfMixing

helmholtzEnergyOfPhaseTransition

helmholtzEnergyOfReaction

helmholtzEnergyOfSolution

helmholtzEnergyOfSolutionAnion

helmholtzEnergyOfSolutionCation

helmholtzEnergyOfSolvation

helmholtzEnergyOfSolvationAnion

helmholtzEnergyOfSolvationCation

helmholtzEnergyOfTransfer

helmholtzSurfaceEnergy

henryCoefficient

hydrationNumber

hydrationNumberAnion

hydrationNumberCation

hydrolysisConstant

idealGasEnthalpy

idealGasGibbsHelmholtzEnergyOfFormationAt25C

idealGasHeatCapacity

ignitionPressureLimit

integralDiffusionCoefficient

integralJouleThomsonCoefficient

integratedParticleFunction

interfacialEnergy

interfacialEnthalpy

interfacialEntropy

interfacialHelmholtzEnergy

interfacialGibbsEnergy

interfacialTension

ionicConductance

ionicConductanceAnion

ionicConductanceCation

ionicMobilityAnion

ionicMobilityCation

ionicRadiusAnion

ionicRadiusCation

ionicStrengthFraction

jouleThomsonCoefficient

kuenenCoefficient

lengthLennardJones

lennardJonesParameter

limitingIonicConductivity

limitingIonicConductivityAnion

limitingIonicConductivityCation

linearExpansionCoefficient

liquidConcentration

liquidConcentrationAnion

liquidConcentrationCation

liquidDensityAt25C

liquidFraction

liquidVolumeAt25C

mass

meanActivityCoefficient

mechanicalSensitivity

meltingPressure

molarPolarization

molarVolume

molecularReactivity

molecularWeight

moles

normalBoilingPoint

normalFreezingPoint

osmoticCoefficient

osmoticPressure

ostwaldAdsorptionCoefficient

parachor

partitionCoefficient

phValue

phaseFraction

phaseTransitionPressure

phaseTransitionTemperature

pkaValue

pkbValue

polarizability

pressure

reactionRateConstant

refractiveIndex

relativeDensity

relaxationTime

salinity

selfDiffusionCoefficient

selfDiffusionCoefficientAnion

selfDiffusionCoefficientCation

solidConcentration

solidSolidPhaseTransitionTemperature

solubility

solubilityCoefficient

solubilityInWater

solubilityParameter(HildebrandAndScott)

solubilityProduct

solvationNumber

solvationNumberAnion

solvationNumberCation

solventDistanceParameter

soundAbsorptionCoefficient

soundVelocity

specificConductivity

specificGravity

stabilityConstant

standardEnthalpyIdealGas

standardEntropy

standardFormationEnthalpy

standardGibbsEnergyOfFormation

standardPotential

sublimationPressure

sublimationTemperature

surfaceArea

surfaceConcentration

surfaceEnergy

surfaceEnthalpy

surfaceEntropy

surfacePressure

surfaceTension

susceptibility

temperature

temperatureOfSolution

thermalConductivity

thermalDiffusivityCoefficient

time

totalComplexPermittivityImaginaryPart

totalComplexPermittivityRealPart

transferActivityCoefficient

transferenceNumberAnion

transferenceNumberCation

transverseSoundVelocity

triplePointPressure

triplePointTemperature

vaporFraction

vaporPressure

virialCoefficient

viscosity

volume

volumeAtStandardConditions

volumeOfMelting

volumeOfMixing

volumeOfTransfer

volumeOfVaporization

volumeSurfaceRatio

waldenProduct

waldenProductAnion

waldenProductCation

watsonKFactor

waveLength

zeroPointEnergy

9.2 Appendix B: Units

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
		no dimension	1.
rad			0.15915475
(BTU.ft3)^(1/2)	(J.m3)^(1/2)		5.4658892
(kJ.m3)^(1/2)	(J.m3)^(1/2)		31.6227766
(J/cm3)^(1/2)	(J/c3)^(1/2)		31.6227766
(BTU/ft3)^(1/2)	(J/m3)^(1/2)		1.9302605E2
(cal/cm3)^(1/2)	(J/m3)^(1/2)		2046.1671
(J/m3)^(1/2)	(J/m3)^(1/2)		1.
(kcal/m3)^(1/2)	(J/m3)^(1/2)		64.70548
(g.l)^(1/2)/min	(kg.m3)^(1/2)/s		1.6666667E-5
(kg.m3)^(1/2)/s	(kg.m3)^(1/2)/s		1.
(lb.ft3)^(1/2)/hr	(kg.m3)^(1/2)/s		3.1481311E-5
(lb.gal)^(1/2)/min	(kg.m3)^(1/2)/s		6.9061846E-4
bar.m6/mol2	Pa.m6/mol2		1.E5
bar.m9.K2/mol3	Pa.m9.K2/mol3		1.E5
bar.m9/mol3	Pa.m9/mol3		1.E5
A	A	Ampere	1.
mA	A		1.E-3
A/cm2	A/m2		10000.
A/m2	A/m2		1.
mA/cm2	A/m2		10.
mA/m2	A/m2		0.001
amagat	amagat		1.
atomic %	atomic fraction		0.01
atomic fraction	atomic fraction		1.
Debye	Coul.m		3.33564E-30
Coul	Coul		1.
Coul.m	Coul.m		1.
Coul/mol	Coul/mol		1.
Farad	Farad		1.
mFarad	Farad		0.001
nFarad	Farad		1.E-9
pFarad	Farad		1.E-12
Farad/m	Farad/m		1.
oz	g		28.349523
H	H		1.

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
H/m	H/m		1.
hr-1	Hz		2.7777778E-4
Hz	Hz		1.
min-1	Hz		1.6666667E-2
s-1	Hz		1.
BTU	J	Britsh thermal unit	1055.0559
cal	J		4.1868
erg	J		1.E-7
eV	J		1.6021892E-19
GJ	J		1.00E9
J	J		1.
kcal	J		4.1868E3
kJ	J		1.E3
kp.m	J		9.80665
kW.hr	J		3.6E6
mJ	J		0.001
MMBTU	J	million BTUs	1.0550559E9
MMkcal	J	million kilocalories	4.1868E9
erg.mK/cm3	J.K/m3		0.1
J.K/m3	J.K/m3		1.
kJ/kg.degC	J.kg.K		1.0E3
J.s	J.s		1.
J/Hz	J.s		1.
J.s/mol	J.s/mol		1.
BTU/cycle	J/cycle		1054.35
cal/cycle	J/cycle		4.1868
GJ/cycle	J/cycle		1.0E9
J/cycle	J/cycle		1.
kcal/cycle	J/cycle		1000.
MMBTU/cycle	J/cycle		1.0550559E9
MMkcal/cycle	J/cycle		4.1868E9
J/kg.degC	J/g.K		1.E-3
J/kg.K	J/g.K		1.E-3
kcal/g.degC	J/g.K		4.1868E3
BTU/degF	J/K	British thermal unit per degrees Fahrenheit	1.899101E3
cal/K	J/K		4.1868
J/K	J/K		1.

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
kcal/K	J/K		4.1868E3
kJ/K	J/K		1.E3
kcal/degC.hr	J/K.s		1.163
BTU/lb	J/kg	Britsh thermal unit per pound avdp.	2.32600E3
cal/g	J/kg		4.1868E3
cal/g	J/kg		4186.8
cal/kg	J/kg		4.1868
J/g	J/kg		1.E3
J/kg	J/kg		1.
kcal(th)/kg	J/kg		4184.
kcal/g	J/kg		4186.8E3
kcal/kg	J/kg		4186.8
kJ/g	J/kg		1.E6
kJ/g	J/kg		1.E6
kJ/kg	J/kg		1.0E3
kW.hr/ton	J/kg		3600.
mJ/kg	J/kg		0.001
MJ/kg	J/kg		1.E6
MMBTU/lb	J/kg	million BTU per pound (avdp)	2.32444E9
MMkcal/kg	J/kg		4.1868E9
BTU/lb.degF	J/kg.K		4.1868E3
BTU/lb.Rnk	J/kg.K		4.1868E3
cal(th)/g.K	J/kg.K		4148.
cal/g.degC	J/kg.K		4.1868E3
cal/g.K	J/kg.K		4186.8
cal/kg.degC	J/kg.K		4.1868
cal/kg.K	J/kg.K		4.1868
J/g.degC	J/kg.K		1.E3
J/g.K	J/kg.K		1.E3
J/kg.K	J/kg.K		1.
kcal/g.K	J/kg.K		4186.8E3
kcal/kg.degC	J/kg.K		4186.8
kcal/kg.K	J/kg.K		4.1868E3
kcal/kg.K	J/kg.K		4186.8
kJ/g.degC	J/kg.K		1.E6
kJ/g.K	J/kg.K		1.E6
kJ/kg.K	J/kg.K		1000.

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
BTU/lb.Rnk2	J/kg.K2		7536.2403
J/kg.K2	J/kg.K2		1.
kcal/kg.K2	J/kg.K2		4186.8
kJ/g.K2	J/kg.K2		1.E6
kJ/kg.K2	J/kg.K2		1000.
BTU/lb.Rnk3	J/kg.K3		13565.2326
kcal/kg.K3	J/kg.K3		4186.8
kJ/g.K3	J/kg.K3		1.E6
kJ/kg.K3	J/kg.K3		1000.
BTU/lb.Rnk4	J/kg.K4		24417.4187
kcal/kg.K4	J/kg.K4		4186.8
kJ/g.K4	J/kg.K4		1.E6
kJ/kg.K4	J/kg.K4		1000.
BTU/lb.Rnk5	J/kg.K5		43951.3537
kcal/kg.K5	J/kg.K5		4186.8
kJ/g.K5	J/kg.K5		1.E6
kJ/kg.K5	J/kg.K5		1000.
J/g.bar	J/kg.Pa		0.01
J/kg.Pa	J/kg.Pa		1.
J/g.bar.K	J/kg.Pa.K		0.01
J/kg.Pa.K	J/kg.Pa.K		1.
J/g.bar2.K	J/kg.Pa2.K		1.E-7
J/kg.Pa2.K	J/kg.Pa2.K		1.
BTU/ft.hr	J/m.s		0.96150757
cal/m.s	J/m.s		4.1868
J/m.s	J/m.s		1.
MMBTU/hr.ft	J/m.s		0.96150757E6
BTU.ft/ft2.hr.Rnk	J/m.s.K		0.09495505
BTU/ft.hr.degF	J/m.s.K		1.730734744
cal.cm/s.cm2.K	J/m.s.K		418.68
kcal/m.hr.degC	J/m.s.K		1.163
MMBTU/hr.ft2.Rnk	J/m.s.K		5.6782633E6
cal/cm2	J/m2		4.1868E4
erg/cm2	J/m2		1.E-3
J/cm2	J/m2		1.E4
J/m2	J/m2		1.
kcal/m2	J/m2		4.1868E3

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
kJ/m2	J/m2		1.E3
BTU/ft2.hr.Rnk	J/m2.K		5.6782636
cal/cm2.K	J/m2.K		4.1868E4
cal/m2.K	J/m2.K		4.1868
J/cm2.K	J/m2.K		1.E4
J/m2.K	J/m2.K		1.
kcal/m2.K	J/m2.K		4.1868E3
kJ/m2.K	J/m2.K		1.E3
BTU/ft2.hr	J/m2.s		3.1545909
BTU/ft2.hr.degF	J/m2.s.K		5.6782636
J/m2.s.K	J/m2.s.K		1.0
BTU/ft3	J/m3		29.875856
cal/cm3	J/m3		4.1868E6
J/m3	J/m3		1.
kcal/m3	J/m3		4.1868E3
kJ/cm3	J/m3		1000.
kJ/m3	J/m3		1.0E3
BTU/ft3.Rnk	J/m3.K		53.776541
cal/cm3.K	J/m3.K		4.1868E6
J/cm3.K	J/m3.K		1000000.
J/m3.K	J/m3.K		1.
kcal/m3.K	J/m3.K		4.1868E3
kJ/m3.K	J/m3.K		1000.
BTU/lbmol	J/mol	British thermal units per pound-moles	2.326E3
BTU/mol	J/mol		1055.056
cal(th)/mol	J/mol		4.184
cal/kmol	J/mol		4.1868E3
cal/mol	J/mol		4.1868
GJ/kmol	J/mol		1.0E12
J/kmol	J/mol		0.001
J/mol	J/mol		1.
kcal(th)/mol	J/mol		4.184E3
kcal/kmol	J/mol		4.1868
kcal/mol	J/mol		4.1868E3
kJ/kmol	J/mol		1.0
kJ/mol	J/mol		1000.
MJ/kmol	J/mol		1000.

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
MMBTU/lbmol	J/mol		2.32444E6
MMkcal/mol	J/mol		4.1868E9
BTU/lbmol.degF	J/mol.K		4.1868
BTU/lbmol.Rnk	J/mol.K		4.1868
BTU/mol.F	J/mol.K		1.8991006E3
BTU/mol.Rnk	J/mol.K		1.8991006E3
cal(th)/mol.K	J/mol.K		4.184
cal/kmol.degC	J/mol.K		4.1868E3
cal/kmol.K	J/mol.K		4.1868E3
cal/mol.degC	J/mol.K		4.1868
cal/mol.K	J/mol.K		4.1868
cal/mol.K	J/mol.K		4.1868
J/kmol.degC	J/mol.K		0.001
J/kmol.K	J/mol.K		0.001
J/mol.degC	J/mol.K		1.
J/mol.K	J/mol.K		1
kcal/kmol.degC	J/mol.K		4.1868
kcal/kmol.K	J/mol.K		4.1868
kcal/mol.degC	J/mol.K		4.1868E3
kcal/mol.K	J/mol.K		4.1868E3
kJ/kmol.degC	J/mol.K		1.
kJ/kmol.K	J/mol.K		1.
kJ/mol.degC	J/mol.K		1000.
kJ/mol.K	J/mol.K		1.E3
BTU/lbmol.Rnk2	J/mol.K2		7.53624
cal(th)/mol.K2	J/mol.K2		4.184
cal/mol.K2	J/mol.K2		4.1868
J/mol.K2	J/mol.K2		1.
kcal/kmol.K2	J/mol.K2		4.1868
kJ/kmol.K2	J/mol.K2		1.
kJ/mol.K2	J/mol.K2		1000.
cal(th)/mol.K3	J/mol.K3		4.184
cal/mol.K3	J/mol.K3		4.1868
J/mol.K3	J/mol.K3		1.
kJ/mol.K3	J/mol.K3		1000.
cal(th)/mol.K4	J/mol.K4		4.184
J/mol.K4	J/mol.K4		1.

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
cal/mol.atm	J/mol.Pa		4.13205E-5
J/mol.Pa	J/mol.Pa		1.
cal/mol.Torr.K	J/mol.Pa.K		0.0314035751
J/mol.Pa.K	J/mol.Pa.K		1.
BTU/hr	J/s		0.29307107
cal/hr	J/s		1.163E-3
cal/s	J/s		4.1868
GJ/hr	J/s		2.7777778E5
J/s	J/s		1.
kcal/hr	J/s		1.1629833
kJ/hr	J/s		0.2777777778
kJ/min	J/s		16.666667
kJ/s	J/s		1.0E3
MJ/hr	J/s		277.77778
MMBTU/day	J/s		1.221129458E4
MMBTU/hr	J/s		0.29307107E6
MMkcal/day	J/s		4.84576375E4
MMkcal/hr	J/s		1.1629833E6
BTU/hr.degF	J/s.K		0.52753056
BTU/hr.Rnk	J/s.K		.527527926
cal/s.K	J/s.K		4.1868
J/s.K	J/s.K		1.
kcal/hr.K	J/s.K		1.1629833
kcal/s.K	J/s.K		4186.8
kJ/hr.degC	J/s.K		0.27777778
kJ/s.degC	J/s.K		1.0E3
kJ/s.K	J/s.K		1.0E3
kcal/hr.m2	J/s.m2		1.1629833
kJ/hr.m2	J/s.m2		0.2777777778
kJ/s.m2	J/s.m2		1000.
kJ/s.m2.K	J/s.m2		1.0E3
kcal/hr.m2.degC	J/s.m2.K		1.1629833
kcal/hr.m2.K	J/s.m2.K		1.1629833
kcal/s.m2.K	J/s.m2.K		4186.8
kJ/hr.m2.degC	J/s.m2.K		0.2777777778
kJ/s.m2.degC	J/s.m2.K		1000.
degC	K	degrees centigrade	1. (+273.15)

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
degF	K		0.55555556 (+255.37222)
K	K		1.
kK	K		1000.
mK	K		0.001
Reamur	K		1.25 (+273.15)
Rnk	K		0.55555556
degF/psia	K/Pa		8.05764E-5
K/atm	K/Pa		9.86923E-6
K/bar	K/Pa		1.E-5
K/MPa	K/Pa		1.E-6
K/Pa	K/Pa		1.
degC-1	K-1		1.
degF-1	K-1		1.8
K-1	K-1		1.
Rnk-1	K-1		1.8
K2	K2		1.
degF2/lbmol2	K2/mol2		1.500111E-6
degF2/lbmol3	K2/mol3		3.3071795E-9
K2/Pa	K2/Pa		1.
K2/Torr	K2/Pa		7.500615E-3
K3	K3		1.
g	kg		0.001
kg	kg		1.
lb	kg	pound avdp.	0.45359237
Mlb	kg	1000 pounds (avdp)	453.59237
ton	kg		1000.
ton(long)	kg		1016.0469
ton(short)	kg		907.18474
kg.m2	kg.m2		1.
lb.in2	kg.m2		2.9263961E-4
g/cycle	kg/cycle		0.001
kg/cycle	kg/cycle		1.
lb/cycle	kg/cycle		0.45359237
Mlb/cycle	kg/cycle		453.59237
ton(short)/cycle	kg/cycle		907.18474
ton/cycle	kg/cycle		1000.
kg/J	kg/J		1.

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
g/100g solvent	kg/kg solvent		0.01
g/kg solvent	kg/kg solvent		0.001
kg/kg solvent	kg/kg solvent	kg solute per kg solvent	1.
mg/kg solvent	kg/kg solvent		1.E-6
kg/m	kg/m		1.
lb/ft	kg/m		1.4881639
kg/m.hr	kg/m.s		2.7777778E-4
slug/hr.ft	kg/m.s		0.013268619
kg/m.hr2	kg/m.s2		7.7160494E-8
kg/m.s2	kg/m.s2		1.
lb/in.s2	kg/m.s2		17.857967
lb/m.s2	kg/m.s2		0.45359237
g.cm2	kg/m2		10.
kg/cm2	kg/m2		1.E-4
kg/m2	kg/m2		1.
lb.ft2	kg/m2		0.04214
poundals/ft2	kg/m2		0.15175047
kg/m2.atm.hr	kg/m2.Pa.s		2.7414535E-9
lb/ft2.atm.hr	kg/m2.Pa.s		1.3384948E-8
g/cm2.s	kg/m2.s		10.
kg/m2.hr	kg/m2.s		2.7777778E-4
kg/m2.s	kg/m2.s		1.
lb/ft2.hr	kg/m2.s		1.3562298E-3
lb/ft2.s	kg/m2.s		4.8824276
g/cm2.s.atm	kg/m2.s.Pa		9.8692326E-5
g/cm2.s.Pa	kg/m2.s.Pa		10.
kg/m2.s.Pa	kg/m2.s.Pa		1.
g/100ml	kg/m3		10.
g/cm3	kg/m3		1000.
g/dm3	kg/m3		1.
g/l	kg/m3		1.
g/m3	kg/m3		1.E-3
g/ml	kg/m3		1.E3
kg/cm3	kg/m3		1.E6
kg/dm3	kg/m3		1.E3
kg/l	kg/m3		1.E3
kg/m3	kg/m3		1.

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
lb/ft3	kg/m3		16.018463
lb/gal	kg/m3		119.82643
lb/in3	kg/m3		27.6799E3
mg/cm3	kg/m3		1.
mg/l	kg/m3		1.E-3
mg/m3	kg/m3		1.E-6
g/100cm3 solvent	kg/m3 solvent		10.
g/l solvent	kg/m3 solvent		1.
kg/m3 solvent	kg/m3 solvent	kg solute per m3 solvent	1.
kg/m3(0 C, 1 atm)	kg/m3(0 C, 1 atm)	kg per norm cubic m	1.
g/cm3.K	kg/m3.K		1.E3
kg/m3.K	kg/m3.K		1.
g/mol	kg/mol		0.001
kg/mol	kg/mol		1.
lb/lb(force).hr.ft2	kg/N.s.m2		3.0489259E-4
g/atm.hr	kg/Pa.s		2.7414535E-6
g/bar.hr	kg/Pa.s		2.7777778E-6
g/kPa.hr	kg/Pa.s		2.7777778E-4
g/kPa.min	kg/Pa.s		1.6666667E-8
g/kPa.s	kg/Pa.s		1.E-6
g/mmHg.hr	kg/Pa.s		2.0835042E-3
kg/atm.hr	kg/Pa.s		2.7414535E-9
kg/atm.s	kg/Pa.s		9.8692327E-6
kg/bar.hr	kg/Pa.s		2.77777778E-9
kg/bar.s	kg/Pa.s		1.E-5
kg/cmH2O.hr	kg/Pa.s		2.8326244E-6
kg/kPa.hr	kg/Pa.s		2.77777778E-7
kg/kPa.min	kg/Pa.s		1.66666667E-5
kg/kPa.s	kg/Pa.s		1.E-3
kg/mmH2O.hr	kg/Pa.s		2.8326244E-7
kg/mmHg.hr	kg/Pa.s		2-0835042E-6
kg/Pa.hr	kg/Pa.s		2.77777778E-4
lb/atm.hr	kg/Pa.s		1.2435024E-9
lb/atm.s	kg/Pa.s		4.4766086E-6
lb/inH2O.hr	kg/Pa.s		5.0585892E-7
lb/inHg(32F).hr	kg/Pa.s		3.7207138E-8
lb/psi.hr	kg/Pa.s		1.827444E-8

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
lb/psi.min	kg/Pa.s		1.0964664E-6
lb/psi.s	kg/Pa.s		6.5787985E-5
g/s	kg/s		1.E-3
kg/day	kg/s		1.157407417E-5
kg/hr	kg/s		0.27777778E-3
kg/min	kg/s		0.0166666667
kg/s	kg/s		1.
lb/day	kg/s		5.249911667E-6
lb/hr	kg/s		0.12599788E-3
lb/s	kg/s		453.59237E-3
Mlb/hr	kg/s	1000 pounds (avdp) per hour	0.12599788
ton/day	kg/s		0.0115741
ton/hr	kg/s		0.27777778
ton/year	kg/s		3.1709792E-5
kg/kW.hr	kg/W.hr		2.777778E-7
lb/hp.hr	kg/W.s		1.689659E-7
kg-1.m-1.s-1	kg-1.m-1.s-1		1.
kJ/cycle	kJ/cycle		1000.
l/Val	l/Val		1.
Ang	m	Angstrøm	1.E-10
cm	m		0.01
dm	m		0.1
ft	m		0.3048
in	m		0.0254
km	m		1000.
m	m		1.
micron	m		1.E-6
mile	m		1609.344
mm	m		1.E-3
nm	m		1.E-9
um	m	mikro-meter	1.E-6
yd	m		0.9144
m/K	m/K		1.
ft/lb	m/kg		6.719690E-1
m/kg	m/kg		1.
bbl/ft2.hr	m/s	barrel per square foot and hour	4.7535474E-4
cm/hr	m/s		2.7777778E-6

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
cm/s	m/s		0.01
cm2/cm.s	m/s		0.01
ft/hr	m/s		8.46666667E-5
ft/min	m/s		5.08E-3
ft/s	m/s		0.3048
ft2/ft.hr	m/s		8.4666667E-5
ft2/ft.s	m/s		0.3048
ft3/ft2.s	m/s		0.3048
ft3/ft2.hr	m/s		8.46666667E-5
gal/ft2.min	m/s		6.79097E-4
km/hr	m/s		0.27777778
km/s	m/s		1000.
l/m2.hr	m/s		2.7778E-7
l/m2.s	m/s		1.E-3
m/hr	m/s		2.77777778E-4
m/min	m/s		0.0166666667
m/s	m/s		1.
m/s	m/s		1.
m2/m.hr	m/s		2.7777778E-4
m2/m.s	m/s		1.
m3/m2.s	m/s		1.
m3/m2.min	m/s		1.6666667E-2
mile/hr	m/s		0.44704
l/hr.rpm	m/s.rpm		2.7777778E-7
m3/min.rpm	m/s.rpm		1.6666667E-2
cm-1	m-1		100.
cm2/cm3	m-1		100.
ft-1	m-1		3.280840
ft2/ft3	m-1		3.2808
in-1	m-1		39.370079
in2/in3	m-1		0.39370079
m-1	m-1		1.
m2/m3	m-1		1.
mm-1	m-1		1000.0
mm2/mm3	m-1		1000.
m-1.s-2	m-1.s-2		1.
cm12/mol4	m12/mol4		1.E-24

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
l4/mol4	m12/mol4		1.E-12
m12/mol4	m12/mol4		1.
cm15/mol5	m15/mol5		1.E-30
m15/mol5	m15/mol5		1.
ft-2	m-2		10.76364864
cm2	m2		1.E-4
dm2	m2		0.01
ft2	m2		0.09290304
in2	m2		0.00064516
m2	m2		1.
mm2	m2		1.E-6
nm2	m2		1.E-18
m2.K/kW	m2.K/W		1.0E-3
m2.K/W	m2.K/W		1.
m2/g	m2/kg		0.001
m2/kg	m2/kg		1.
ft2/lb(force)	m2/N		2.0885434E-2
m2/Ohm	m2/Ohm		1.
cm2/Ohm.mol	m2/Ohm.mol		1.E-4
m2/Ohm.mol	m2/Ohm.mol		1.
Sie.cm2/mol	m2/Ohm.mol		0.01
cm2/Ohm.val	m2/Ohm.val		1.E-4
m2/Ohm.val	m2/Ohm.val		1.
bbl/ft.hr	m2/s	barrel/(foot and hour)	9.1134442E-4
cm2/s	m2/s		1.E-4
cSt	m2/s	centistokes	1.E-6
ft2/hr	m2/s		2.58064E-5
ft2/min	m2/s		1.548385E-3
ft3/ft.hr	m2/s		2.58064E-5
l/hr.m	m2/s		2.7777778E-7
m2/day	m2/s		1.1574074E-5
m2/hr	m2/s		2.7777778E-4
m2/s	m2/s		1.
m2/year	m2/s		3.1688087E-8
m3/m.min	m2/s		1.6666667E-2
m3/m.s	m2/s		1.
mm2/s	m2/s		1.E-6

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
mSt	m2/s		1.E-7
St	m2/s		1.E-4
cm2/s.mol fraction	m2/s.mol fraction		1.
m2/s.mol fraction	m2/s.mol fraction		1.
m2/s2	m2/s2		1.
cm2/V.s	m2/V.s		1.E-4
m2/V.s	m2/V.s		1.
bbl	m3	barrel	0.15898729
cm3	m3		1.0E-6
dm3	m3		1.E-3
ft3	m3		2.8316847E-2
gal	m3		3.7854118E-3
in3	m3		1.6387064E-5
l	m3	liters	1.E-3
m3	m3		1.
ml	m3		1.E-6
cm3.K/mol	m3.K/mol		1.E-6
ft3.Rnk/lbmol	m3.K/mol		3.468220043E-5
m3.K/kmol	m3.K/mol		1.E-3
bbl/cyle	m3/cycle	barrel/cycle	0. 15898729
ft3/cycle	m3/cycle		2.8316847E-2
gal/cycle	m3/cycle		3.7854118E-3
kbbl/cycle	m3/cycle	1000 barrels per cycle	0.15898729
l/cycle	m3/cycle		0.001
m3/cycle	m3/cycle		1.
MMft3/cycle	m3/cycle	million cubic feet per cycle	2.8316847E4
kbbl/day	m3/day		1.840131E-6
cm3/kg	M3/g		1.E-3
m3/K	m3/K		1.
cm3/100g	m3/kg		1.E-5
cm3/g	m3/kg		1.E-3
dm3/g	m3/kg		1.
dm3/kg	m3/kg		1.E-3
ft3/lb	m3/kg		0.062427962
l/g	m3/kg		1.
l/kg	m3/kg		1.E-3
m3/kg	m3/kg		1.

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
ml/g	m3/kg		1.E-3
cm3/g solvent	m3/kg solvent		1.E-3
m3/kg solvent	m3/kg solvent		1.
m3/kg.s	m3/kg.s		1.
m3/kg.s2	m3/kg.s2		1.
cm3/cm3 solvent	m3/m3 solvent		1.
cm3/l solvent	m3/m3 solvent		1.E-3
l/l solvent	m3/m3 solvent		1.
m3/m3 solvent	m3/m3 solvent		1.
ml/ml solvent	m3/m3 solvent		1.
cm3/cm3(sat.solut.)	m3/m3(sat.solution)		1.
cm3/l.atm	m3/m3.Pa		0.0098692327E-6
m3/m3.Pa	m3/m3.Pa		1.
dm3/mol	m3/mol		1.E-3
ft3/lbmol	m3/mol		6.242796029E-5
l/mol	m3/mol		1.E-3
m3/kmol	m3/mol		0.001
m3/mol	m3/mol		1.
ml/mol	m3/mol		1.E-6
cm3/mol2	m3/mol2		1.E-6
cm3/bar.g	m3/Pa.kg		1.E-8
m3/Pa.kg	m3/Pa.kg		1.
bbl/psi.hr	m3/Pa.s	barrel/(pounds per square inch and hour)	6.4053188E-9
ft3/psia.min	m3/Pa.s		6.8450163E-8
ft3/psia.s	m3/Pa.s		4.107010E-6
l/atm.hr	m3/Pa.s		2.7414535E-12
l/kPa.hr	m3/Pa.s		2.7777778E-10
l/mmHg.hr	m3/Pa.s		2.0835042E-9
m3/kPa.hr	m3/Pa.s		2.7777778E-7
m3/kPa.min	m3/Pa.s		1.6666667E-5
m3/kPa.s	m3/Pa.s		1.E-3
ml/atm.hr	m3/Pa.s		2.7414535E-15
ml/kPa.hr	m3/Pa.s		2.7777778E-13
bbl/day	m3/s	barrel/day	1.8401307E-6
bbl/hr	m3/s	barrel/hour	0.04415E-3
ft3/day	m3/s		3.2774128E-7
ft3/hr	m3/s		7.8657907E-6

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
ft3/min	m3/s		4.7194744E-4
ft3/s	m3/s		0.028316846
gal/hr	m3/s		1.0515033E-6
gal/min	m3/s		6.3090196E-5
l/day	m3/s		1.157407E-8
l/hr	m3/s		2.7777778E-7
l/min	m3/s		1.6666667E-5
l/s	m3/s		0.001
l/year	m3/s		3.1688087E-11
m3/day	m3/s		1.1574074E-5
m3/hr	m3/s		2.7777778E-4
m3/min	m3/s		1.6666667E-2
m3/s	m3/s		1.
m3/year	m3/s		3.1688087E-8
Mft3/day	m3/s	1000 cubic feet per day	0.327741279E-3
MMft3/day	m3/s	million cubic feet per day	0.327741279
MMft3/hr	m3/s	million cubic feet per hour	7.8657907
cm3/hr.K	m3/s.K		2.7777778E-10
l/min.m	m3/s.m		1.6666667E-5
l/s.m	m3/s.m		0.001
m3/hr.m	m3/s.m		2.7777778E-4
l/min.m2	m3/s.m2		1.6666667E-5
l/s.m2	m3/s.m2		0.001
m3/hr.m2	m3/s.m2		2.7777778E-4
ft3/hr.psi	m3/s.Pa		1.140836E-9
ft3/min.rpm	m3/s.rpm		4.7194744E-4
ft3/s.rpm	m3/s.rpm		0.028316846
gal/hr.rpm	m3/s.rpm		1.0515033E-6
gal/min.rpm	m3/s.rpm		6.3090196E-5
l/day.rpm	m3/s.rpm		1.157407E-8
l/min.rpm	m3/s.rpm		1.6666667E-5
l/s.rpm	m3/s.rpm		0.001
m3/day.rpm	m3/s.rpm		1.1574074E-5
m3/hr.rpm	m3/s.rpm		0.2777778E-3
m3/s.rpm	m3/s.rpm		1.
m3/year.rpm	m3/s.rpm		3.1688087E-8
ft3/s2	m3/s2		0.028316846

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
m3/s2	m3/s2		1.
cm2.l/Ohm.mol2	m5/Ohm.mol2	square cm*liters per Ohm and mole squared	1.E-7
m5/Ohm.mol2	m5/Ohm.mol2		1.
cm2.l/s.mol	m5/s.mol		1.E-7
m5/s.mol	m5/s.mol		1.
cm6/mol2	m6/mol2		1.E-12
dm6/mol2	m6/mol2		1.E-6
ft6/lbmol2	m6/mol2		3.8972502E-9
l2/mol2	m6/mol2		1.E-6
m6/mol2	m6/mol2		1.
cm9/mol3	m9/mol3		1.E-18
dm9/mol3	m9/mol3		1.E-9
ft9/lbmol3	m9/mol3		2.4329738E-13
l3/mol3	m9/mol3		1.E-9
m9/mol3	m9/mol3		1.
mass fraction	mass fraction		1.
kmol	mol		1000.
lbmol	mol	pound-mole	453.592368
mol	mol		1.
mol %	mol fraction		1.E-2
mol fraction	mol fraction		1.
mol/mol	mol fraction		1.
kmol/cycle	mol/cycle		1000.
lbmol/cycle	mol/cycle		453.59237
mol/cycle	mol/cycle		1.E-3
lbmol/day	mol/day		5.249911667E-3
lbmol/day.ft	mol/day.m		1.722412E-2
mol/100g	mol/kg		10.
mol/kg	mol/kg		1.
molon	mol/kg		1.
umol/100 g	mol/kg	micro-mol per 100 grams	1.E-5
mol/m.min	mol/m.s		1.6666667E-2
mol/m.s	mol/m.s		1.
dmol/m3	mol/m3		0.1
kmol/m3	mol/m3		1.E3
lbmol/ft3	mol/m3		1.6018463E4
lbmol/gal	mol/m3		1.1982643E5

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
mmol/cm3	mol/m3		0.001
mmol/l	mol/m3		1.
mol/cm3	mol/m3		1.E6
mol/dm3	mol/m3		1.E3
mol/l	mol/m3		1.E3
mol/m3	mol/m3		1.
mol/l solvent	mol/m3 solvent		1000.
mol/m3 solvent	mol/m3 solvent		1.
mol/m3.K	mol/m3.K		1.
kmol/m3.atm	mol/m3.Pa		9.86923E-3
mol/m3.atm	mol/m3.Pa		9.86923E-6
mol/m3.Pa	mol/m3.Pa		1.
lbmol/ft3.hr	mol/m3.s		4.4495732
lbmol/ft3.min	mol/m3.s		2.6697439E2
lbmol/ft3.s	mol/m3.s		1.6018563E4
mol/cm3.hr	mol/m3.s		3.6E9
mol/cm3.s	mol/m3.s		1.E6
mol/l.hr	mol/m3.s		0.277777778
mol/l.s	mol/m3.s		1.E3
mol/mol solvent	mol/mol solvent		1.
lbmol/atm.s	mol/Pa.s		0.0044766086
lbmol/psi.min	mol/Pa.s		1.0964664E-3
lbmol/psi.s	mol/Pa.s		6.5787985E-2
mol/atm.hr	mol/Pa.s		2.7414535E-9
mol/bar.hr	mol/Pa.s		2.7777778E-9
mol/kPa.hr	mol/Pa.s		2.7777778E-7
mol/kPa.min	mol/Pa.s		1.66666667E-5
mol/kPa.s	mol/Pa.s		1.E-3
kmol/day	mol/s		1.157407417E-5
kmol/hr	mol/s		0.27777778
kmol/s	mol/s		0.001
lbmol/hr	mol/s		0.12599788
lbmol/s	mol/s		0.45359237E3
mol/min	mol/s		1.6666667E-2
mol/s	mol/s		1.E-3
kmol/day.m	mol/s.m		1.157407417E-5
kmol/hr.m	mol/s.m		0.27777778E-6

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
kmol/s.m	mol/s.m		0.001
lbmol/hr.ft	mol/s.m		0.41337887
lbmol/s.ft	mol/s.m		1.4881639E3
kmol/hr.m3	mol/s.m3		0.27777778E-6
kmol/s.m3	mol/s.m3		0.001
kmol/hr.atm	mol/s.Pa		2.7414535E-6
kmol/hr.bar	mol/s.Pa		2.77777778E-6
kmol/hr.kPa	mol/s.Pa		2.77777778E-4
kmol/hr.mmH2O	mol/s.Pa		0.0283262437
kmol/hr.Pa	mol/s.Pa		0.27777778
kmol/min.kPa	mol/s.Pa		0.0166666667
kmol/s.atm	mol/s.Pa		9.8692327E-9
kmol/s.bar	mol/s.Pa		1.E-8
kmol/s.kPa	mol/s.Pa		1.E-6
mol-1	mol-1		1.
mol2/l2	mol2/m6		1.E6
mol2/m6	mol2/m6		1.
mol/100g solvent	molal		10.
mol/g solvent	molal		1000.
mol/kg solvent	molal		1.
molal	molal		1.
molal-1	molal-1		1.
molar	molar		1.
dyn	N		1.0E-5
lb(force)	N		4.4482216
N	N		1.
dyn^(1/4).cm^(11/4)/ml	N^(1/4).m^(11/4)/mol		1.7782794E-7
N^(1/4).m^(11/4)/mol	N^(1/4).m^(11/4)/mol		1.
lb(force).ft	N.m		1.3558179
lb(force).ft/lb	N.m/kg		2.989067
lb(force).ft/s	N.m/s		1.3558179
mN.s/m2	N.s/m2		0.001
dyn/cm	N/m		1.E-3
lb(force)/ft	N/m		14.593903
mN/m	N/m		1.E-3
N/m	N/m		1.
N/m2.m	N/m		1.

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
lb(force)/ft2	N/m2		47.880259
lb(force)/ft3	N/m3		157.08747
mmH2O/m	N/m3		9.806348
N/m3	N/m3		1.
Ohm	Ohm		1.
Ohm.cm	Ohm.m		0.01
Ohm.m	Ohm.m		1.
Ohm-1	Ohm-1		1.
Ohm-1.cm-1	Ohm-1.m-1		100.
Ohm-1.m-1	Ohm-1.m-1		1.
Sie/cm	Ohm-1.m-1		100.
Sie/m	Ohm-1.m-1		1.
Ohm-1.m-3	Ohm-1.m-3		1.
Sie/ml	Ohm-1.m-3		1.E6
ata	Pa	absolute atmospheres	98066.5
atm	Pa		101325.
bar	Pa		1.E5
cmH2O	Pa	cm of water column (pressure)	98.063754
Gpa	Pa		1.E9
Hpa	Pa		100.
inH2O	Pa	inches of water columns (pressure)	249.0889
inHg(32F)	Pa		3386.39
inHg(60F)	Pa		3332.69
kbar	Pa		1.E8
kg(force)/cm2	Pa		98066.5
kN/m2	Pa		1000.
kp/cm2	Pa		98066.5
kp/m2	Pa		9.80665
kPa	Pa		1.E3
lb(force)/in2	Pa		6.89476E3
lb/in2	Pa		6.89476E3
mbar	Pa		100.
mmH2O	Pa	mm of water column	9.806348
mmHg	Pa	mm of Hg column (Torr)	133.3223684
MN/m2	Pa		1.E6
mPa	Pa		1.E-3
MPa	Pa		1.E6

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
mTorr	Pa		0.1333224
N/m2	Pa		1.
Pa	Pa		1.
psi	Pa	pounds per square inch	6894.7573
Torr	Pa		133.3224
ata.K	Pa.K		98066.5
atm.K	Pa.K		101325.
kPa.K	Pa.K		1000.
Pa.K	Pa.K		1.
kg(force).m/kg	Pa.m/kg		9.80665
bar.m/s	Pa.m/s		1.E5
Pa.m/s	Pa.m/s		1.
atm.ft3/lbmol	Pa.m3/mol		6.3255133
Pa.cm3/mol	Pa.m3/mol		1.E6
Pa.dm3/mol	Pa.m3/mol		1.E3
Pa.m3/mol	Pa.m3/mol		1.
psi.ft6	Pa.m6		5.5285184
atm.dm6.K/mol2	Pa.m6.K/mol2		0.101325
Pa.m6.K/mol2	Pa.m6.K/mol2		1.
atm.l2.K2/mol2	Pa.m6.K2/mol2		0.101325
bar.m6.K2/mol2	Pa.m6.K2/mol2		1.E5
kPa.m6.K2/mol2	Pa.m6.K2/mol2		1000.
Pa.m6.K2/mol2	Pa.m6.K2/mol2		1.
bar.cm6/g.K	Pa.m6/kg.K		1.E2
Pa.m6/kg.K	Pa.m6/kg.K		1.
atm.l2/mol2	Pa.m6/mol2		0.101325
kPa.m6/mol2	Pa.m6/mol2		1000.
Pa.m6/mol2	Pa.m6/mol2		1.
psi.ft6/lbmol2	Pa.m6/mol2		2.68706E-5
psi.ft9	Pa.m9		0.15655021
kPa.m9.K2/mol3	Pa.m9.K2/mol3		1000.
kPa.m9/mol3	Pa.m9/mol3		1000.
Pa.m9/mol3	Pa.m9/mol3		1.
psi.ft9/lbmol3	Pa.m9/mol3		1.6774764E-9
kbar.mol.K/cm3	Pa.mol.K/m3		1.E14
Pa.mol.k/m3	Pa.mol.K/m3		1.
cP	Pa.s	centipoise	0.001

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
g/cm.s	Pa.s		0.1
kg(force).s/m2	Pa.s		9.80665
kg/m.s	Pa.s		1.
kp.s/m2	Pa.s		9.807
lb(force).s/ft2	Pa.s		47.880262
lb/ft.hr	Pa.s		4.1337887E-4
lb/ft.s	Pa.s		1.48816
mP	Pa.s		1.E-4
mPa.s	Pa.s		1.E-3
N.s/m2	Pa.s		1.
P	Pa.s		0.1
Pa.s	Pa.s		1.
psi.hr	Pa.s		2.4821136E7
ug/cm.s	Pa.s	mico-gram per cm and s	1.E-7
uP	Pa.s	micropoise	1.E-7
uPa.s	Pa.s	micro-Pascal-seconds	1.E-6
mPa.s.cm3/g	Pa.s.m3/kg		1.E-6
Pa.s.m3/kg	Pa.s.m3/kg		1.
uPa.s.cm3/g	Pa.s.m3/kg	micro-Pascal.s.cm3/g	1.E3
Pa.s/K	Pa.s/K		1.
Pa.s/K2	Pa.s/K2		1.
Pa.s/K3	Pa.s/K3		1.
cP/mol %	Pa.s/mol fraction		0.1
cP/mol fraction	Pa.s/mol fraction		0.001
Pa.s/mol fraction	Pa.s/mol fraction		1.
lb/ft.hr2	Pa.s2		1.1482746E-7
lb/ft.s2	Pa.s2		1.4881662
ata/K	Pa/K		98066.5
atm/K	Pa/K		101325.
bar/K	Pa/K		1.E5
kbar/K	Pa/K		1.E8
kPa/K	Pa/K		1.E3
mbar/K	Pa/K		1.E2
mPa/K	Pa/K		1.E-3
MPa/K	Pa/K		1.E6
Pa/K	Pa/K		1.
psi/F	Pa/K		12410.568

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
Torr/K	Pa/K		133.3224
ata/m	Pa/m		98066.5
atm/m	Pa/m		101325.
bar/km	Pa/m		100.
bar/m	Pa/m		1.E5
cmH2O/m	Pa/m		98.063754
inH2O/ft	Pa/m		817.2208
inHg(32F)/ft	Pa/m		11110.203
kPa/km	Pa/m		1.E6
kPa/mm	Pa/m		1.
mbar/m	Pa/m		1.E2
mH2O/m	Pa/m		98.063754
mmHg/ft	Pa/m		437.4094
Pa/m	Pa/m		1.
psi/100ft	Pa/m		2.2620595E2
psi/ft	Pa/m		2.2620595E4
Torr/ft	Pa/m		437.40945
kbar/cm3.mol	Pa/m3.mol		1.E14
Pa/m3.mol	Pa/m3.mol		1.
ata/mol fraction	Pa/mol fraction		98066.5
atm/mol fraction	Pa/mol fraction		101325.
bar/mol fraction	Pa/mol fraction		1.E5
kPa/mol fraction	Pa/mol fraction		1.E3
MPa/mol fraction	Pa/mol fraction		1.E6
Pa/(mol/mol)	Pa/mol fraction		1.
Pa/mol fraction	Pa/mol fraction		1.
psi/mol fraction	Pa/mol fraction		6894.76
Torr/mol fraction	Pa/mol fraction		133.3224
ata/s	Pa/s		98066.5
atm/s	Pa/s		101325.
bar/s	Pa/s		1.E5
Pa/s	Pa/s		1.
atm/weight fraction	Pa/weight fraction		101325.
Pa/weight fraction	Pa/weight fraction		1.
MPa^0.5	Pa^0.5		1.E3
Pa^0.5	Pa^0.5		1.
ata-1	Pa-1		1.01972E-5

unit	SI-unit	remarks	factor unit -> SI-unit (only for information)
atm-1	Pa-1		9.86923E-6
bar-1	Pa-1		1.E-5
cm2/dyn	Pa-1		10.
cm2/kg(force)	Pa-1		1.0197162E-5
cm2/kp	Pa-1		1.01972E-5
Gpa-1	Pa-1		1.E-9
inH2O-1	Pa-1		4.0146309E-3
kbar-1	Pa-1		1.E-8
kPa-1	Pa-1		1.E-3
m2/kp	Pa-1		0.101972
m2/N	Pa-1		1.
mbar-1	Pa-1		1.E-2
mmH2O-1	Pa-1		0.1019748
mPa-1	Pa-1		1.E3
MPa-1	Pa-1		1.E-6
Pa-1	Pa-1		1.
psi-1	Pa-1		1.4503774E-4
Torr-1	Pa-1		7.500615E-3
TPa-1	Pa-1		1.E-12
atm-1.K-1	Pa-1.K-1		9.86923E-6
Pa-1.K-1	Pa-1.K-1		1.
P-1	Pa-1.s-1		10.
Pa-1.s-1	Pa-1.s-1		1.
ata-2	Pa-2		1.0398289E-10
atm-2	Pa-2		9.7401753E-11
bar-2	Pa-2		1.E-10
kPa-2	Pa-2		1.E-6
mbar-2	Pa-2		1.E-4
Pa-2	Pa-2		1.
Torr-2	Pa-2		5.6259226E-5
ata-3	Pa-3		1.0603225E-15
atm-3	Pa-3		9.6128057E-16
bar-3	Pa-3		1.E-15
kPa-3	Pa-3		1.E-9
mbar-3	Pa-3		1.E-6
Pa-3	Pa-3		1.
Torr-3	Pa-3		4.219788E-7

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
ata-4	Pa-4		1.081228E-20
atm-4	Pa-4		9.4871016E-21
bar-4	Pa-4		1.E-20
kPa-4	Pa-4		1.E-12
mbar-4	Pa-4		1.E-8
Pa-4	Pa-4		1.
Torr-4	Pa-4		3.1651005E-9
day	s		86400.
hr	s		3600.
min	s		60.
ms	s		0.001
ns	s		1.E-9
ps	s		1.E-12
s	s		1.
us	s	microseconds	0.000001
year	s		31536000.
hr.ft2.Rnk/BTU	s.m2.K/J		1.7611019E-1
hr.m2.K/kcal	s.m2.K/J		8.5984523E-1
s.cm2.K/cal	s.m2.K/J		2.3884590E-5
s.m2.K/J	s.m2.K/J		1.0
s.m2.K/kcal	s.m2.K/J		2.3884590E-4
s.m2.K/kJ	s.m2.K/J		1.0E-3
day-1	s-1		1.1574074E-5
GHz	s-1		1.E9
kHz	s-1		1000.
MHz	s-1		1000000.
rad/s	s-1		0.15915475
year-1	s-1		3.1688088E-8
kV	V		1.E3
mV	V		1.E-3
V	V		1.
V/cm	V/m		100.
V/m	V/m		1.
val/kg	val/kg		1.
val/l	val/m3		1.E3
val/m3	val/m3		1.
cm3/100cm3	volume fraction		0.01

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
cm3/cm3	volume fraction		1.
cm3/l	volume fraction		1.E-3
l/m3	volume fraction		1.E-3
m3/m3	volume fraction		1.
ml/m3	volume fraction		1.E-6
volume %	volume fraction		0.01
volume fraction	volume fraction		1.
BTU/s	W		1.0550559E3
erg/s	W		1.E-7
hp	W	hourse power	745.69987
kcal/s	W		4.1868E3
kp.m/s	W		9.80665
kW	W		1000.
PS	W		745.700
W	W		1.
hp.hr	W.s		2.6845195E6
kW/cycle	W/cycle		1000.
W/cycle	W/cycle		1.
W/degC	W/K		1
W/K	W/K		1.
W/kg	W/kg		1.
W/m	W/m		1.
BTU.in/ft2.hr.degF	W/m.K		0.14422789
BTU.in/ft2.hr.Rnk	W/m.K		0.14422789
BTU/ft.hr.degF	W/m.K		1.7307347
BTU/ft.s.degF	W/m.K		6230.6449
BTU/in.hr.degF	W/m.K		20.768816
BTU/in.s.degF	W/m.K		7.4767738E4
cal/cm.s.K	W/m.K		4.1868E2
cal/km.s.K	W/m.K		4.1868E-3
cal/m.hr.K	W/m.K		1.163E-3
cal/m.s.K	W/m.K		4.1868
erg/cm.s.K	W/m.K		1.E-5
J/cm.s.K	W/m.K		1.E2
J/m.s.K	W/m.K		1.
kcal.m/hr.m2.K	W/m.K		1.163
kcal/m.hr.K	W/m.K		1.163

unit	**SI-unit**	**remarks**	**factor unit -> SI-unit (only for information)**
kcal/m.s.K	W/m.K		4.1868E3
kW/m.K	W/m.K		1.0E3
mW/cm.degC	W/m.K		0.1
mW/cm.K	W/m.K		0.1
mW/m.K	W/m.K		0.001
uW/cm.K	W/m.K	micro-Watt per cm and K	0.0001
W/cm.K	W/m.K		1.E2
W/m.K	W/m.K		1.
W/m.K2	W/m.K2		1.
W/m.K3	W/m.K3		1.
W/m.K4	W/m.K4		1.
W/m2	W/m2		1
cal/cm2.s.K	W/m2.K		4.1868E4
kW/m2.K	W/m2.K		1.0E3
W/m2.degC	W/m2.K		1.
W/m2.K	W/m2.K		1.
W/m2.s.K	W/m2.s.K		1.
hp/ft3	W/m3		2.6334143E4
kW/l	W/m3		1.E6
kW/m3	W/m3		1.E3
W/m3	W/m3		1.
W/mol	W/mol		1.
hp/lbmol.hr	W/mol.s		4.566631EE-4
hp/lbmol.s	W/mol.s		1.64398709
kW/kmol.hr	W/mol.s		2.7777778E-4
kW/kmol.s	W/mol.s		1.
g/100g	weight fraction		1.E-2
g/g	weight fraction		1.
g/kg	weight fraction		0.001
kg/kg	weight fraction		1.
mg/kg	weight fraction		1.E-6
ppm	weight fraction		1.E-6
weight %	weight fraction		1.E-2
weight fraction	weight fraction		1.

bbl	= petroleum barrel, USA (42 US liquid gallons)
cal(th)	= thermochemical calory

gal	= liquid gallons, USA
hp	= mechanical horse power
lb	= pound (avdp.)
oz	= ounces (avdp.)
u...	= mikro...
m...	= milli...
M...	= Mega...

9.3 Appendix C: Models or equations

This Appendix contains a selection of model equations which may be part of a PPDB. This selection is not meant to be exclusive, every PPDB can add its own equations.

Predictive equations like UNIFAC are exclude on purpose, because they do not need any model parameters.

short name of method	**full name of method**	**equation**	**parameters**
Antoine	Antoine vapor pressure equation	$\log(p) = a - b/(T+c)$	a, b, c
Wrede	Wrede vapor pressure equation	$\log(p) = a - b/T$	a, b
Wrede-ln	Wrede vapor pressure equation	$\ln(p) = a - b/T$	a, b
Cragoe	Cragoe vapor pressure equation	$\log(p) = a + b/T + c*T + d*T^2$	a, b, c, d
Riedel	Riedel vapor pressure equation	$\ln(p) = a - b/T + c*T + d*T^2 + e*\ln(T)$	a, b, c, d, e
Wagner	Wagner vapor pressure equation	$\ln(p/p_{crit}) = (a*x + b*x^{(3/2)} + c*x^3 + d*x^6)/(T/T_{crit})$; $x = 1 - T/T_{crit}$	a, b, c, d, criticalPressure, criticalTemperature
Wagner2	2nd Wagner vapor pressure equation	$\ln(p/p_{crit}) = (a*x + b*x^{(3/2)} + c*x^3 + d*x^7 + e*x^9)/(T/T_{crit})$; $x = 1 - T/T_{crit}$	a, b, c, d, e, criticalPressure, criticalTemperature
Wagner3	Wagner vapor pressure equation (Aspen)	$\ln(p/p_{crit}) = (a*x + b*x^{(3/2)} + c*x^3 + d*x^4)/(T/T_{crit})$; $x = 1 - T/T_{crit}$	a, b, c, d, criticalPressure, criticalTemperature
Chebyshev	Chebyshev vapor pressure equation	$T*\log(p) = c_0/2 + \sum^{(s)}[c_s * E_s(x)]$; $x = 2*T - (T_{max}+T_{min})/(T_{max}-T_{min})$; $E_s(x)$ = Chebyshev polynomial of order s	c_0, c_1, c_2, ..., T_min, T_max
polynomial	polynomial	$y = a + b*x + c*x^2 + ...+ j*x^9$; x = any property	a, b, c, d, e, f, g, h, i, j
vapor pressure_1	vapor pressure equation	$\ln(p) = a + b*T + c/T + d/T^2$	a, b, c, d
mod.Antoine(Hysys)	modified Antoine vapor pressure equation (Hysys[9], page A-36)	$\ln(p) = A + B/(T+C) + D*T + E*\ln(T) + F*T^G$	A, B, C, D, E, F, G

short name of method	full name of method	equation	parameters
mod.Antoine(Aspen)	modified Antoine vapor pressure equation (Aspen[7], page 3-80)	$\ln(p) = A + B/(T+C) + D*\ln(T) + E*T^F$	A, B, C, D, E, F
Jones-Dole	Jones-Dole equation	$\eta/\eta_0 = 1 + a*\sqrt{c} + b*c$	a, b, viscosity_0
Yen-Woods	Yen-Woods equation for densities	$d = d_{crit} * (a + \Sigma^{(j)}(k_j)*(1-T/T_{crit})^{(j/3)})$	criticalDensity, criticalTemperature, a, k_0, k_1, k_2, ...
Antoine viscosity	Antoine equation for the viscosity	$\ln(\eta) = a + b/(T+c)$	a, b, c
Riedel therm.cond.	Riedel equation for thermal conductivities	$\kappa = a * (1 + (20/3)*(1 - T/T_{crit})^{(2/3)})$	a, criticalTemperature
Sprow/Prausnitz	Surface Tension after Sprow and Prausnitz	$\sigma = a * (1 - T/T_{crit})^b$	a, b, criticalTemperature
modified polynomial	modified polynomial	$property = a + b/T + c/T^2 + d*T + e*T^2 + f*T^3 + ...$	a, b, c, d, e, f, ...
Yuan/Mok	Yuan - Mok equation for the heat capacity	$c_p = a + b * \exp(-c/T^n)$	a, b, c, n
Redlich-Kister	Redlich-Kister equation for excess properties in binary systems	$\Delta\ property = x_1 * x_2 * \Sigma^{(i)}(a_i*(x_1-x_2)^{i})$	a_0, a_1, a_2, a_3, ...
thermal conductivity (NEL)	NEL equation for thermal conductivity	$\kappa = a*(1 + b*x^{(1/3)} + c*x^{(2/3)} + d*x)$; $x=1-T/T_{crit}$	a, b, c, d, criticalTemperature
virial equation	virial equation	$Z = 1 + vc_2*p + vc_3*p^2 + vc_4*p^4 + ...$	vc_2, vc_3, vc_4
BWR	BWR-equation of state	$p = R*T*d + (b_0*R*T - a_0 - c_0/T^2)*d^2 + (b_0*R*T - a_0)*d^3 +a*\alpha*d^6 + (c*d^3/T^2)*(1 + \gamma*d^2)*\exp(-\gamma*d^2)$	a_0, b_0, c_0, a, b, c, alpha, gamma
BWR-Lee-Starling	Benedict-Webb-Rubin-Lee-Starling equation of state (Aspen[7], page 3-8)	$Z_m = Z_{m0} + \gamma_i*Z_{m1}$; $Z_{m0}, Z_{m1} = function(T,T_{crit},v_m,v_{crit,m})$	criticalTemperature _i, criticalVolume_i, gamma_i, epsilon_i_j, eta_i_j
Hayden-O'Connel	Hayden-O'Connel equation of state (Aspen[7], page 3-9)	$Z_m = 1 + B*p/R*T$; $B = \Sigma^{(i)}\Sigma^{(j)}B_{ij}(T)$	B_i_j
Lee-Kesler	Lee-Kesler equation of state (Aspen[7], page 3-18)	$Z = Z_0 + (Z_r-Z_0)*\omega/\omega_r$; $Z_0 = fct_0(T/T_{crit}, p/p_{rit})$; $Z_r = fct_r(T/T_{crit}, p/p_{crit})$	criticalTemperature, criticalPressure, omega
Lee-Kesler-Plöcker	Lee-Kesler-Plöcker equation of state (Aspen[7], page 3-19)	$Z_m = Z_{m0} + (\omega/\omega_r)*(Z_{m0} - Z_{mr})$; $Z_{m0} = fct_0(T,T_{crit}, v_m, v_{crit,m})$; $Z_{mr} = fct_r(T,T_{crit},v_m,v_{crit,m})$ mixing rules for $v_{crit,m}$, T_{crit}	criticalTemperature, criticalPressure, vcriticalVolume, omega, Z_c_i, K_i_j
Peng-Robinson-Boston-Mathias	Peng-Robinson-Boston-Mathias equation of state (Aspen[7], page 3-25)	$p = R*T/(v_m-b) - a/[v_m*(v_m+b)+b*(v_m-b)]$	criticalTemperature_i, criticalPressure_i, omega_i, k_1_2
Redlich-Kwong	Redlich-Kwong equation of state (Aspen[7], page 3-27)	$p = R*T/(v_m-b) - (a/\sqrt{(T)})/[v_m*(v_m+b)]$	criticalTemperature_i, criticalPressure_i

short name of method	full name of method	equation	parameters
Redlich-Kwong-Aspen	Aspen modification of the Redlich-Kwong equation of state(Aspen[7], page 3-28)	$p = R*T/(v_m-b) - a/[v_m*(v^m+b)]$ with mixing rules	criticalTemperature_i, criticalPressure_i, omega_i, eta_i,k_0_a__i_j, k_1_a_i_j, k_0_D_i_j, k_1_D_i_j,
Redlich-Kwong-Soave-Boston-Mathias	Redlich-Kwong equation of state with Boston-Mathias alpha function (Aspen[7], page 3-29)	$p = R*T/(v_m-b) - a/[v_m*(v_m+b)]$ with mixing rules	criticalTemperature_i, criticalPressure_i, omega_i, k_i_j
Schwartzentruber-Renon	Schwartzentruber-Renon equation of state (Aspen[7], page 3-31)	$p = R*T/(V_m+c-b) - a/[(v_m+c)*(V_m+c+b)]$ with mixing rules	criticalTemperature_i, criticalPressure_i, omega_i, q_0_i, q_1_i, q_2_i, c_0_i, c_1_i, c_2_i, k_0_a_i-j, k_1_a_i_j, k_2_a_i_j, l_0_i_j, l_1_i_j, l_2_i_j, k_0_D_i_j, k_1_D_i_j, k_2_D_i_j
Peng-Robinson	standard Peng-Robinson equation of state (Aspen[7], page 3-34)	$p = R*T/(v_m-b) - a/[v_m*(v_m+b)+b*(v_m-b)]$	criticalTemperature_i, criticalPressure_i, omega_i (i=1..2), k_1_2
Redlich-Kwong-Soave	standard Redlich-Kwong-Soave equation of state (Aspen[7], page 3-35)	$p = R*T/(v_m-b) - a/[v_m*(v_m+b)]$ with mixing rules	criticalTemperature_i, criticalPressure_i, omega_i, k_i_j
Bromley-Pitzer	Bromley-Pitzer activity coefficient model (Aspen[7], page 3-54)		beta_ion, delta_ion, beta_0, beta_1, beta_2, beta_3
Chien-Null	Chien-Null model for calculation activity coefficient of highly non-ideal systems (Aspen[7], page 3-55)		a_i_j, b_i_j, v_i_j
Electrolyte-NRTL	NRTL activity coefficient model for electrolytes(Aspen[7], page 3-58)		A_B, B_B, C_B, r_i, A_BB, A_BsB, B_BBs, B_BsB, alpha_BB, F_BBs, F_BsB, G_BBs, G_BsB, C_ca_B, C_B_ca, D_ca_B, D_B_ca, E_ca_B, E_B-ca, alpha_ca_B, C_cas_cass, C_cass-cas, c_csa,cssa, C_cssa_csa, D_cas_cass,D_cass_cas, D_csa_cssa, D_cssa_csa, E_cas_cass, E_cass_cas,E_csa_cssa, E_cssa_csa, alpha_cas_cass, alpha_csa_cssa
NRTL	NRTL activity coefficient model (DDB[8], page XVI)		A_i_j, A_j_i, alpha (i,j=1...2)
extended NRTL (Aspen)	NRTL activity coefficient model (Aspen[7], page 3-62)		a_i_j, b_i_j, c_i_j, d_i_j, e_i_j, f_i_j (i,j=1...2)
general NRTL	general NRTL activity coefficient model (Hysys[9], page A-22)		form-of_equation, A_j_j, B_i_j, C_i_j, F_i_j, G_i_j, alpha1_i_j, alpha2_i_j (i,j=1...2)
Pitzer activity	Pitzer model for activity		beta_0, beta_1, beta_2,

short name of method	**full name of method**	**equation**	**parameters**
coefficient model	coefficients of aqueous systems (Aspen [7], page 3-63)		beta_3, C_p, theta_c_cs, theta_a_as, psi_c_cs_a, psi_c_a_as
Redlich-Kister	Redlich-Kister model for calculating activity coefficients (Aspen[7], page 3-66)		a_i_j, b_i_j, c_i_j, d_i_j, e_i_j, f_i_j, g_i_j, h_i_j, m_i_j, n_i_j, v_i
Scatchard-Hildebrand	Scatchard-Hildebrand model (Aspen[7], page 3-67)		criticalTemperature-i, delta_i, V_i_CVT, V_i_l
Margules	Margules equation for calculating liquid activity coefficients (DDB[8], page XVI)	$\ln(\gamma_i) = [A_{ij} + 2*(A_{ji}-A_{ij})*x_i](1-x_i)^2$	A_i_j
extended Margules	Margules equation for calculating liquid activity coefficients with temperature-independent parameters (Hysys[9], page A-24)	$\ln(\gamma_i) = (1-x_i)*+2*[A_i + 2+x_i*(B_i-A_i)]$; $A_i = \Sigma^{(j)}[x_j*(a_{ij}+b_{ij}*T)/(1-x_i)]$; $B_i = \Sigma^{(j)}[x_j*(a_{ji}+b_{ji}*T)/(1-x_i)]$	a_i_j, b_i_j (i,j=1...2)
three-suffix Margules	extended Margules equation for calculating liquid activity coefficients (Aspen[7], page 3-68)		a_i_j, b_i_j, c_i_j, d_i_j (i,j=1...2)
van Laar	van Laar equation for calculating liquid activity coefficients (DDB[8], page XVI)		A_i_j (i,j=1...2)
extended van Laar (Aspen)	extended van Laar equation for calculating liquid activity coefficients with temperature-independent parameters (Aspen[7], page 3-75)		a_i_j, b_i_j, c_i_j, d_i_j (i,j=1...2)
extended van Laar (Hysys)	extended van Laar equation for calculating liquid activity coefficients with temperature-independent parameters (Hysys[9], page A-28)		a_i_j, b_i_j (i,j=1...2)
Wilson	Wilson equation for calculating liquid activity coefficients (DDB[8], page XVI)		A_i_j (i,j=1...2)
extended Wilson (Aspen)	extended Wilson equation for calculating liquid activity (Aspen[7], page 3-78)		a_i_j, b_i_j, c_i_j, d_i_j (i,j=1...2)
extended Wilson (Hysys)	extended Wilson equation for calculating liquid activity coefficients with temperature-independent parameters (Hysys[9], page A-29)		a_i_j, b_i_j (i,j=1...2)
UNIQUAC	UNIQUAC equation for calculating liquid activity coefficients (DDB[8], page XVII)		u_i_j (i,j=1...2)
extended UNIQUAC (Aspen)	extended UNIQUAC equation for calculating liquid activity coefficients with temperature-independent parameters (Aspen[7], page 3-74)		a_i_j, b_i_j, c_i_j, d_i_j (i,j=1...2)

short name of method	**full name of method**	**equation**	**parameters**
extended UNIQUAC (Hysys)	extended UNIQUAC equation for calculating liquid activity coefficients with temperature-independent parameters (Hysys[9], page A-26)		a_i_j, b_i_j (i,j=1...2)
DIPPR107	DIPPR equation for the ideal heat capacity	property = $A + B[C/T/\sinh(C/T)]^2 + D[E/T/\cosh(E/T)]^2$	A, B, C, D, E
heat capacity (ASPEN)	Aspen[7]-equation for the solid heat capacity (page 3-102)	$C_p = c_1 + c_2{*}T + c_3{*}T^2 + c_4/T + c_5/T_2 + c_6/\sqrt{(T)}$	c1, c2, c3, c4, c5, c6
Barin	Barin equations for thermophysical property data	$G = a + b{*}T + c{*}(T{*}\ln(T)) + d{*}T^2 + e{*}T^3 + f{*}T^4 + g/T + h/T^2$	a, b, c, d, e,f, g, h
Andrade	Andrade equation for calculating the liquid viscosity	$\ln(\eta) = A + B/T + C{*}\ln(T)$	A, B, C
liquid viscosity (DIPPR)	DIPPR equation for the liquid viscosity	$\ln(\eta) = c_1 + c_2/T + c_3{*}\ln(T) + c_4{*}T^{c5}$	c1, c2, c3, c4, c5
viscosity mixing rule	ASPEN[7] mixing rule for the liquid viscosity (listed under the heading Andrade/DIPPR, page 3-122)	$\ln(\eta) = \Sigma^{(i)}[x\_i{*}\ln(\eta_i)] + \Sigma^{(i,j)}[(a_{ij} + b_{ij}/T){*}x_i{*}x_j + (c_{ij}+d_{ij}/T){*}x_i^2{*}x_j^2]$	a_i_j, b_i_j, c_i_j, d_i_j
DIPPR102	DIPPR equation for the gas viscosity at 0 atm pressure and the gas thermal conductivity	property = $A{*}T^B/(1 + C/T + D/T^2)$	A, B, C, D
Chung-Lee-Starling	Chung-Lee-Starling correlation of the viscosity and thermal conductivity of liquid or gaseous mixtures (Aspen[7], page 3-127, 3-138))		criticalTemperature_i, V_crit_i, dipole_moment_i, omega_i, kappa_i, xi_i_j, zeta_i_j
suface tension (DIPPR)	DIPPR correlation for surface tension	$\sigma = c1{*}(1-T_r)\char`^(c_2 + c_3{*}T_r + c_4{*}T_r^2 + c_5{*}T_r^3)$; $T_r = T/T_{crit}$	c1, c2, c3, c4, c5, criticalTemperature
Hakim-Steinberg-Stiel	Hakim-Steinberg-Stiel equation for the surface tension (Aspen[7], page 3-155)		chi
DIPPR105		property = $A/B\char`^[1+(1-T/C)^D]$	A, B, C, D
DIPPR101		property = $\exp(A + B/T + C{*}\ln(T) + D{*}T^E)$	A, B, C, D, E
DIPPR106		property = $A{*}(1-T^r)\char`^(B + C{*}T_r + D{*}T_r^2)$; $T_r = T/T_{crrit}$	A,B,C,D, criticalTemperature
DIPPR104		property = $A + B/T + C/T^3 + D/T^8 + E/T^9$	A, B, C, D, E

Glossary of the symbols used in the column "equation"

cp	heat capacity
d	density
d_{crit}	critical density

p	pressure
p_{crit}	critical pressure
R	gas constant
T	temperature
T_{crit}	critical temperature
v_m	volume of a mixture
x_1	mole fraction of compound 1
Z	compressibility factor
Z_m	compressibility factor of a mixture
γ_i	activity coefficient of compound i
κ	thermal condictivity
η	viscosity
η_0	viscosity at zero concentration
σ	surface tension

9.4 Appendix D: Phase equilibrium information

This list is considered to be preliminary. A common list valid for all CAPE-OPENn applications will be developed by another CAPE-OPEN committee.

(An extension to Chapter 2.15.1 of "Open Interface Specifications Thermodynamic and Physical Properties" CO-THRM-1 Version 1.07[6])

VaporLiquid	Vapor-liquid equilibrium
LiquidLiquid	Liquid-liquid equilibrium
VaporLiquidLiquid	Vapor-liquid equilibrium with two liquid phases
LiquidSolid	Solid-liquid equilibrium
SolidSolid	Solid-solid equilibrium
LiquidSolidSolid	Solid-liquid equilibrium with two solid phases
VaporLiquidSolid	triple point, 3 coexisting phases: vapor, liquid and solid
VaporSolid	Solid-vapor equilibrium, sublimation
Overall	no phase equilibrium present

Unknown

Emulsion

Suspension

SinglePhase

Azeotrope

Eutectic

9.5 Appendix E: state of aggregation

This list is considered to be preliminary. A common list valid for all CAPE-OPENn applications will be developed by another CAPE-OPEN committee.

Vapor	Gaseous phase
Liquid	Liquid phase
Solid	Solid phase
Liquid-1	Liquid phase #1
Liquid-2	Liquid phase #2
Liquid-3	
Liquid-4	
Liquid-5	
Liquid-6	
Liquid-7	
Liquid-8	
Liquid-9	
Solid-1	Solid phase #1
Solid-2	Solid phase #2
Solid-3	
Solid-4	
Solid-5	
Solid-6	
Solid-7	

Solid-8	
Solid-9	
Fluid	supercritical gas or liquid
LightLiquid	see "liquid"
HeavyLiquid	see "liquid"
Unknown	
VaporLiquidInterface	
LiquidLiquidInterface	
LiquidSolidInterface	

9.6 Appendix F: Table information, model parameter set information

isobar

isochore

isocomposition

isotherm

9.7 Appendix G: Additional Property Specifications

Binode	LLE, data are on the binodal curve
TieLine	LLE, points are linked by a tie line
recommended	Recommended values, carefully evaluated data
rejected	
estimated	
measured	
smoothed	