Methods and Tools
Special Interest Group Report
CAPE-OPEN 2017 Annual Meeting
Sunbury-on-Thames

Bill Barrett

US Environmental Protection Agency

12 October, 2017

SIG Membership

Bill Barrett
US EPA

Jasper van Baten
AmsterCHEM

Tony Garratt
Reaction Design

Daniel Wagner
DWSIM

Mark Stijnman
Shell Global Solutions

Jorge Martinis and
Michael Hlavinka
Bryan Research & Engineering, LLC

Loic d'Anterroches
Céondo GmbH

David Jerome and
Krishna Penukonda
Schneider Electric

M&T SIG Ongoing Activities

- Common Interface conference calls
  - First Wednesday of the month
- Flowsheet Monitoring conference call
  - Second Wednesday of the month.
- Threading conference call (short term)
  - Third Wednesday of the month
- Object Model conference call
  - Fourth Wednesday of the month
- Join? Please contact either SIG Leader or CTO
  - Bill Barrett – barrett.williamm@epa.gov
  - Michel Pons - technologyofficer@colan.org

M&T SIG Charter

- Improve integration, and expand utilization of Computer-Aided Process Engineering (CAPE) applications within the enterprise through identification and resolution of existing cross-cutting issues with the CAPE-OPEN platform, develop mechanisms for use of CAPE within other application domains, and incorporate advances in information technology into the CAPE-OPEN platform.

- Key responsibilities
  - Resolve issues with the common interface specifications.
  - Develop and maintain standards and protocols for CAPE-OPEN implementations.
  - Incorporate advances in information technology into the CAPE-OPEN protocols.
  - Identify novel uses of CAPE and provide standards for utilizing CAPE within these applications.

No Change to vision and responsibilities.

M&T SIG 2017/2018 Activities

- **Common Interface Specification:**
  - Utilities Errata and Clarifications document
  - Error Handling issues
  - Development of Flowsheet Monitoring Interface Specification

- **Support of Interoperability SIG:**
  - Type Library installer and .NET Primary Interop Assembly.

- **One-Time Request:**
  - Quad-precision floating-point values

- **COBIA Development**

Utilities Common Interface

- **STATUS of Errata and Clarifications Document**
  - Peer review completed
  - Thank you to reviewers who submitted feedback on the document.
  - M&T SIG will finalize based on comments received.
  - Request to Management Board to publish by 2017 end.

- **Clarifications (Reminder):**
  - Identifies Primary PMC Objects that must implement *ICapeUtilities*
  - Edit Method Return Value:
    - 0 = S_OK: PMC was modified
    - 1 = S_FALSE: PMC remains unmodified
  - Simulation Context must be set prior to *ICapeUtilities.Initialize* for PMEs that provide Simulation Context.
    - *ICapeDiagnostic* to be available to PMC once Simulation Context is set.
  - PMC Object Life Cycle steps enumerated.

Error Common Interface

- Issues identified:
  - Complexity
    - CAPE-OPEN error handling not based on COM *ErrorInfo* API.
    - CAPE-OPEN objects are required to expose all possible CAPE-OPEN error interfaces
    - Most CAPE-OPEN objects only expose *ECapeRoot* and *ECapeUser* and return *ECapeUnknownHR*
    - Error conditions are not transparent.
  - Logging tools are required to identify the cause of problems.
- Errata and Clarification Document unlikely
- Need to document the COBIA approach.

Flowsheet Monitoring Interface

- New Interface Specification
  - Provides the ability to access all elements in a flowsheet without interfering with the flowsheet
  - Can respond to event notifications for modification of flowsheet configuration.
- Draft will be completed and sent to CAPE-OPEN membership for peer review.
- Prototypes and demonstration will be developed.

- Michel Pons will distribute document to members for peer review this year.

Quad Precision Floating Point

- Request from CO-LaN member to determine options to support quad precision in thermodynamic calculations.
- Current CAPE-OPEN uses double (R8) precision floating-point values.
- Use of quad precision is growing in scientific applications.
- Non-standard extended precision-real values have historically existed.
- IEEE Standard for Floating-Point Arithmetic (IEEE 754 -2008) was revised in 2008 to include quadruple precision (128-bit) floating point values.
  - *Two 128-bit formats: binary and decimal*
  - *Octuple (256-bit) precision also specified*
- Current relevant IEEE 754-2008 Implementations:
  - Hardware
    - Intel Power9 CPU
  - Software
    - Intel and GNU FORTRAN
    - MATLAB
    - BOOST

Quad Precision Floating Point, cont'd

- Options identified for use of quad precision:
  - Option 1: No CAPE-OPEN Support at the moment
    - Microsoft COM and Visual Studio compilers currently do not support quad precision.
  - Option 2: Creation of a shadow set of quad-precision interfaces
    - Duplicate interfaces where CapeQuad replaces CapeDouble in methods.
    - Support for 64-bit double precision would be required minimum.
  - Option 3: Use platform-specific default
    - Replace CapeDouble data type with CapeReal in interface specifications.
    - CapeReal would be 64-bit (R8) on platforms without 128-bit support, or binary128 on platforms that support that format.
    - Calls between different platforms can be marshaled to convert types.
- Adopting Options 2 or 3 would require modification of textual interface specification documents.
- At present, Option 1 applies.

COBIA Roadmap

- Phase I – Proof of Concept
  - Core technical components
  - Demonstrate COM/COBIA interoperability with Thermo 1.1 interface set
- Phase II – Full Windows Native
  - Expanding COBIA to all interfaces of business value
  - Support for C/C++ development.
  - Allow development of fully functional COBIA-based PMEs and PMCs
- Phase III – Cross Platform Interoperability (Future)
  - Microsoft .NET is planned
  - Other platforms as identified by CO-LaN membership

COBIA Timeline

- **October 2016 - Phase I completed**
- **October 2017 – Phase II status presented and demonstrated**
- **2017/8**
  - Finalize design decisions for, and complete development of COBIA Phase II
  - Revise Common Interface Specifications
    - This will incorporate issues raised in the published Errata and Clarifications documents.
  - Work with other SIGs to transition to COBIA
    - Likely minor modifications to interface specifications documents
- **FUTURE**
  - .NET language bindings (Committed to development)
  - Other language bindings developed as needed.
  - CO-LaN will maintain COBIA codebase and provide updates as needed.
  - COBIA Training.

COBIA 2017/2018 Activities

- Develop Persistence Interface Specification
- New Parameters Interface Specification
- New Reporting Interface Specification
- Error Handling
- Threading Model
- Query to Thermodynamics and Unit SIGs Regarding inclusion of Thermodynamics 1.0 Interfaces.

Proposal for New Persistence Common Interface

- **Goals**
  - Less ambiguity than COM persistence.
  - Use of platform-native serialization mechanisms.
    - No special developer knowledge required, e.g. .NET development should only require use of Serializable attribute.
    - Developers can still customize persistence.
    - Use of human-readable formats such as JSON or XML
- **Design decision**
  - Explicit separation of object serialization from storage to persistent media.
  - COMBIA handles COM persistence interoperability.
  - Limited number of persistence interfaces.
    - *ICapePersist* – Exposed by the PMC
    - *ICapePersistReader* and *ICapePersistWriter* – Implemented by the PME.
- **Status - Proposal submitted.**

Persistence Common Interface, Cont'd.

- ***ICapePersist Methods:***
  - ***Save*** **– Asks the PMC to save itself using the *ICapePersistWriter* interface.**
  - ***Restore*** **– Restores the PMC using the *ICapePersistReader* interface.**

- ***ICapePersistWriter***
  - **Exposed by PME persistence Object.**
  - **Provides methods to write standard CAPE-OPEN data types and byte array.**

- ***ICapePersistReader***
  - **Exposed by PME persistence Object.**
  - **Provides methods to read standard CAPE-OPEN data types and byte array.**

Proposal for Parameter Common Interface

- Proposed interfaces are similar to current Parameter interfaces, with the following modifications:
- Goal - Strongly typed Value/Elimination of the VARIANT.
- Separation of common and type-specific aspects of the parameter.
  - Common Parameter properties:
    - Mode
    - Type
    - Validate/Validation Status
  - Type-specific Parameter properties:
    - Value
    - Upper/Lower Bounds
    - Default Value
    - Units of Measure only provided for Real and Real Array parameters
- Backwards compatible with existing COM-based CAPE-OPEN implementations provided by COMBIA interoperability.

PARAMETER Common Interface, cont'd

- Array Parameters:
  - Parameter value will be CapeArray(type)
    - CapeArrayInteger
    - CapeArrayReal
    - CapeArrayBoolean
    - CapeArrayString
  - Can obtain and set the value of the entire array.
  - Parameter values can be obtained/set by element using the ICapeArray(Type)Parameter interface.
- ICapeArrayParameter interface has two properties:
  - NumDimensions
  - Size
- NOTE: All rows have the same length, not a jagged array.

- STATUS: Still converging on design specification.

Proposed Reporting Common Interface

- Issues leading to updating Reporting Interface:
  - Currently Reporting is handled by ICapeUnitReport
    - Reporting is only supported by unit operation PMCs.
    - Making it separate would encourage use by other PMCs.
    - PMEs could ask any PMC whether it supported reporting.
  - Format of reports available is currently limited.
    - Work around using Simulation Context.
- Proposal submitted to develop ICapeReport Interface.
  - Use Cases:
    - Provide a list of available reports.
    - Provide a list of available report formats.
    - Generate the desired report in the desired output format.
  - Report formats:
    - Possible new formats: HTML, PDF, Rich Text
    - Allow inclusion of images (svg, gif)
    - Potential for XML/JSON format that can be used in templated reporting.
- This interface is currently not anticipated to be available in COM-based CAPE-OPEN.

Development of Thread Model

- **Objective – Enable efficient use of COBIA in multi-threaded applications.**
- **Context:**
  - **Availability of multiple threaded hardware is increasing as costs are decreasing.**
  - **Multiple threads may improve application performance.**
  - **Multi-threaded applications are becoming more prevalent.**
  - **Most CAPE applications are tightly coupled making thread safety difficult to ensure.**
- **Constraints:**
  - **Legacy components were not developed with thread safety in mind.**
    - **Evolution of programming standards – standardization of thread safety methods is a relatively new requirement.**
    - **Modifiable Global variables, shared resources, etc.**
  - **Microsoft COM apartment threading is widely used in CAPE-OPEN.**
  - **Development of thread safe software is difficult.**
- **Status – Community Input required on possible solutions.**

Scenario for Multi-Threaded PMC

- Distillation column unit operation PMC
  - Each tray has a material object representing the contents of the tray.
  - Calculation of PMC can be split amongst multiple threads.
  - This PMC updates the overall state of these internal Material Objects.
  - The PMC then invokes a flash on all the Material Objects using different threads.
  - The Material Objects then call the flash routine in the Equilibrium Servers.

- Involves calls from Unit Operation PMC to the PME's Material Object, then to a third-party Property Package PMC.
  - Could be three separate objects from three different vendors/developers.
  - Need to ensure that these calls do not block each other or cause race conditions in either the PME or PMCs.

COBIA Licensing

- M&T SIG reviewed a number of licenses (MIT, BSD, Apache, GPL) to develop a general overview of licensing issues associated with COBIA.
- Desirable features:
  - Does not require modifications to code to be released by modifier (no copyleft type provisions).
  - Allows use, sale, and redistribution of source and binaries.
  - Third party issues – CO-LaN indemnification.
  - Protects CO-LaN from liability.
  - CO-LaN not responsible for model accuracy.
  - No warranty/As-is provision.
  - Limit the use of CO-LaN logo/No endorsement from CO-LaN for use.
  - Require acknowledgement of COBIA use.
  - Retains CO-LaN's ability to establish standards for CAPE-OPEN compliance.
- Status – Advice provided to Management Board.

2017/8 Deliverables

- **Errata and Clarifications documents**
  - Utilities – Finalize peer review and publish

- **Flowsheet Monitoring interface:**
  - Finalize document
  - Peer review
  - Publish

- **COBIA:**
  - Complete Phase II development and distribute.
  - Revised Error Common interfaces
  - Revised Parameter Common interfaces
  - Revised Persistence Common interfaces
  - New Reporting Common interfaces

Disclaimer

The views expressed in this presentation are those of the authors and do not necessarily reflect the views or policies of the U.S. Environmental Protection Agency.