Object-oriented modelling applied to hybrid unit operations

Paolo Greppi, Barbara Bosio, Elisabetta Arato
Department of Civil Environmental Architectural Engineering
Università degli Studi di Genova

ICheaP-8 - The eight International Conference on Chemical & Process Engineering
ISCHIA Island Gulf of Naples, Italy - June 24-27th 2007
Contents

- Introduction
- Streams, flashes, connectivity, multistage arrangements
- Applications
- Conclusions
Introduction: model workflow

library developer

LIBPF LIBRARY

model developer

model executable

model user
Model development

- map process simulation entities to
  - abstract C++ class types
  - mixins: reusable partial classes
  - concrete classes

= LIBrary for Process Flowsheeting

- pick up concrete classes from LIBPF
- write a C++ program representing the system model
- compile it to a standalone executable
Model deployment

- receives system model as an executable
- access via **user interface**
- can change inputs or switch configuration
- each model configuration has a fixed structure
Contents

- Introduction
  - Streams, flashes, connectivity, multistage arrangements
- Applications
- Conclusions
The simplest object: \texttt{stream}

- abstract \texttt{stream} type
- specialized mixin classes are derived from abstract \texttt{stream} type
Concrete stream types

- every concrete stream type contains at least one total phase
- multi-phase streams can have explicit representations of the phases
  - stream_VLe...
  - stream_VLLSSSe...
Enter reactions

- Take a generic flash (genflash) mixin class instance
- add reactions ...
- ... get reactive flashes
Enter connectivity

Special mixin types represent the capability of units to connect to streams
Compose mixins ...

connectivity mixin

process mixin
.. get a concrete class
Variations of genflash with connectivity
Combinations of generic flashes

- flexible mulstream concentrated parameters model: multihx
- N reactive streams exchanging heat or mass:
  - (reactive) multi-stream heat exchanger
  - (reactive) membrane unit
  - fuel cell
Multistage arrangements

- Concentrated parameter objects are combined to yield distributed parameter models:
  - 1-D arrangements:
    - columns
    - pipes
  - 2-D arrangements:
    - fuel cells
    - reactive heat exchangers
Contents

- Introduction
- Streams, flashes, connectivity, multistage arrangements
- Applications
- Conclusions
Application 1: Simple distillation column

The column was broken down into 8 subunits:

1. splitter
2. mixer
3. \texttt{multistage1D<genflashNX<stream_VL>}}
4. \texttt{genflashNX<stream_VL>}}
Application 2: Distributed parameter planar Fuel Cell

- 5 x 10 multihx objects

flowsheet for cross-flow arrangement
Application 2: Distributed parameter planar Fuel Cell

- Results match proven tools
- Example: solid temperature plot for MCFC (Molten Carbonate Fuel Cell)
Applications: simulation run duration

LIBPF slower than special-purpose tools

<table>
<thead>
<tr>
<th>Test case</th>
<th>Reference tool (benchmark)</th>
<th>Benchmark Timing, s</th>
<th>LIBPF Timing, s</th>
</tr>
</thead>
<tbody>
<tr>
<td>Simple distillation column</td>
<td>Commercial process simulator X</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>Planar Fuel Cell</td>
<td>Special purpose 2-D finite differences code</td>
<td>2</td>
<td>60</td>
</tr>
</tbody>
</table>
Applications: modelling project duration

modelling of hybrid / complex unit operations

- conventional approach:
  3 man-months

- new approach with model reuse:
  1 man-month

(estimates)
Conclusions

- **C++** can be used for process simulation using **LIBPF** library.
- **Run-time** is slower than with conventional approaches ...
- ... but **project duration** is shorter for hybrid unit operations modelling.
Visit **libpf.com**

- Get C++ header files with the **classes hierarchy**
- Get and run **demos**
- **Request** the LIBPF library by mail