CHEMICAL ENGINEERING TRANSACTIONS

VOL. 32, 2013

A publication of AIDIC
The Italian Association of Chemical Engineering
Online at: www.aidic.it/cet

Chief Editors: Sauro Pierucci, Jiří J. Klemeš
Copyright © 2013, AIDIC Servizi S.r.l.,
ISBN 978-88-95608-23-5; **ISSN** 1974-9791

DOI: 10.3303/CET1332235

The Role of Couplers in Chemical, Model-Based Engineering

Heinz A. Preisig[a*], Stephen C. Lobo [b]

[a]Dept of Chemical Engineering, NTNU, Trondheim, Norway
[b]Dept of Material Science and Engineering, NTNU, Trondheim, Norway
Heinz.Preisig@chemeng.ntnu.no

Interdisciplinary computational engineering work implies the use of different programs taken from different disciplines and performing a series of concerted computations. The user is thereby faced with the task to bring together different code that has not been designed to interact with each other. Rather they are stand-alone pieces of software that require information and that produce some well-define results. Couplers provide the infrastructure to bring such pieces to talk together and form a computation environment assembled for a specific purpose.

1. The Stage

Computer-based engineering uses models of parts of the real world to predict its behaviour. This model thus acts as a surrogate of the process of interest. Models are the core of science and engineering comprising the main body of knowledge besides data that reflects the behaviours of specific objects. Formulating models is a non-trivial matter and in itself an essential activity. A model is built for a purpose, a need that has been formulated first. The model must fit this purpose or else it has to be adjusted. It should be simple, but must, as one habitually says, be just complex enough -- complex enough to capture the characteristics that are essential to meet the defined need.
The same models are used in different contexts and since the solution methods are often developed in the context domain, there is a need for making models transportable and exchangeable. The exchange is not only a matter of not having to redo them thus being able to inherit and save on effort, but to maintain consistency, to indeed use the same model in different applications and contexts.
Many domains have developed solutions to domain-generic problems, for example in the field of fluid-mechanics many solvers are available, a field commonly known as computation fluid mechanics. Whilst these packages have built-in a part of the model, usually something to the extent of Navier-Stokes equations and continuity equations they do not have the physical properties integrated and must import them through an interface. A classic case of a model, namely the thermodynamic material behaviour description that must be imported and that should be the same for all studies associated with the process in question.
In engineering there is often the need of having a library of sub-processes so as to be able to assemble more complex structures from simple ones, like a box of Lego, in which one can build objects from principle building blocks and where one can combine also larger objects to even more magnificent structures. In chemical process systems engineering this is the old lady called flow-sheeting package in which as part of the last evolution, graphical interfaces were constructed that allow the assembling of flowsheets for larger structures building on models for process units.
In nanotechnology, there is currently a need formulated that takes models ranging from the smallest, sub-molecular scale all the way to the macroscopic engineering scale. This integration draws resources from very many different fields including quantum chemistry, molecular dynamics, micro-fluids, meso-scale population-based behaviour simulations all the way to macroscopic flows and capacity behaviour, plant and control design. The latter clearly uses computational procedures from different communities and requires multiple-domain expertise

2. Technology Overview

The domain of model and data exchange has a long history. Some of the main efforts pertinent to chemical engineering are CapeOpen (CoLan, 2012) and pdXi, which stands for Process Data eXchange Institute an effort established by late Rodolphe Motard of Washington University in Saint Louis in 1985. The US effort was aiming at generating a hub for process models, but has failed to be supported by the leading companies (Blaha, 2010). There has been a remainder in the form of a company operating under Blaha in the US. The efforts are driven by the recognition at that a lot of the unit models can be re-used and also strongly driven by the need of consistent thermodynamic material descriptions.
In recent years similar problems become apparent in the Modelica society, within which an effort called Modelisar an ITEA2 European project in which the functional Mock-up Interface (FMI) was developed.
In the France, CERFACS developed PalmOpen (2011), whilst in the US, the Sandia Lab developed LIME. In Germany the Frauenhofer institute has a product called MpCCI (2012), TISC (2012). These last mention efforts are “couplers”.

2.1 Cape-Open

The motivation for the project was to bring together the chemical industry and the software vendors to generate tools that better facilitate model exchange. The targeted market was primarily the flowsheeting community, which has been dominated by a very few players in the market, a situation that has little changed since the project got funded. The project had two main target areas, namely enabling the exchange of unit models and facilitating an interface for thermodynamic data providers.
Whilst the objectives were very honourable, each party was protecting its own interests and the best compromise for all parties involved took the form of wrappers. Also, one of the main constraints was that the community wanted to enable the use of legacy code aiming at preserving well-established functioning code. The wrapper is a combination of a shell code written in C^{++}, a CORBA layer which are compiled together and then linked. The integration of the shell and the object code of the core code is a non-trivial step which often requires manual adjustments.
Cape-Open is a worthy tool that enhances portability of process engineering models by making the individual model components less dependent on the coding environment. The legacy code is wrapped in contrast to be modified putting the effort into the wrapper instead of modifying the model. However this also has the effect of disguising the model and the modelled unit’s behaviour is lost behind layers of wrapping code.
The major drawbacks of Cape-Open are the wrapping itself as it introduces the extra wrapping and compiling, the increase of the overhead for interacting with the model. Secondly, the Cape-Open standard does not implement any requirements on the model itself. This means it is up to the user to determine how to fit the model into the Cape-Open framework. In some cases, vendors have created Cape-Open compliant modules, but this is a commercial solution and further details on implementation are protected. The details behind the interface are also protected by the flowsheeting package vendors. They are not part of the Cape-Open standard. The standard provides an appropriate compromise for the handling of existing models, however makes no attempt to suggest innovative ways for building models in the first place. Instead it focuses on the end user by creating an easy “plug and play” solution.

2.2 The Process Data Exchange Institute

In 1989, a proposal for addressing the problem of data exchange received the support of the Computers & Systems Technology Division of the American Institute of Chemical Engineers (AIChE) and in 1991 the Process Data Exchange Institute (pdXi) was formed under AIChE auspices. This was the result of work done by Rodolphe Motard’s group at Washington University, St Louis (NSF, 1985). In 1995 the institute reported 23 sponsoring companies supporting the pdXi’s objectives to establish and maintain open approaches to electronically exchange and manage process data between computer applications, databases and organisations with PSE. The spent effort was significant and pdXi reported having defined over 500 objects and 1300 attributes for their characterisation. Today, these documents published on this effort are no longer available. The institute has ceased to exist quite some time ago indicating the lack of interest and the inability to come up with a strong unifying concept that goes beyond enumerating objects (PDXI, 1995).

2.3 Multi-Physics Couplers

The term “multi-physics” reflects the concerted use of several codes each of which may be solving a physical modeling problem. Thus “multi-physics” merely reflect the fact that several separate “packages” are involved and coupled together in one or the other way.

The interaction of different tasks is in no way any new invention, but has been around for a long time. At least as long as manufacturers have implemented real-time operating systems in which it inter-tasks communication is a basic functionality (see for example the RSX-11 of DEC’s PDP-11 operating system). It was also one of the driving forces behind the development of Petri-nets, namely logically coordinated multi-tasking.
So task-task communications are one of the core activities being enabled in couplers. One of the common mechanisms being used is the message passing system, for which the Argonne national labs developed a standard: Message-Passing Interface (MPI). The most recent standard was published by the MPI Forum in Sept 2012. An alternative standard is provided by OpenMP, (2013).
Couplers, though, do more: They provide means of controlling tasks, thus spawn tasks and stop and exit tasks. They enable interactions with data bases and tools that help to do auxiliary computations for example to match one model’s outputs with the other model’s input. An example for the latter is the matching of two codes that use a gridded representation of distributed systems, where the grids are not matching exactly on the boundary of the two distributed domains each being simulated in a different code.
Couplers enable con-current execution and intercommunication. Couplers also provide the services for inter-task communications for programs that have not been specially designed for this purpose, some auxiliary computations, dynamic control of the programs typically support for parallel computing. In order to deal with the inevitable problem of a multi-language environment, wrapper technologies like SWIG are being used to construct wrappers for legacy code.

There exist a good number of couplers: CERFACS developed PalmOpen (2011), the Sandia Lab developed LIME (a sourceforge open source project). In Germany the Frauenhofer institute has a product called MpCCI (2012), and a company tlk-thermo has a product called TISC (2012).

2.4 Functional Mock-Up Interface (FMI)

The FMI is the product of the MODELISAR project of ITEA2 Europe (MFI, 2013). It is “a tool-independent standard to support both model exchange and co-simulation of dynamic models using a combination of xml-files and compiled C-code.” It is supported by a good number of tools. Currently 35 are in the list.
The core of the FMI is the creation of Functional Mockup Units (FMUs). To facilitate model exchange, the units are designed so that equations, variable and events (time, state and step) are easily accessible to the modeling environment. The interface deals with ordinary differential equations in continuous state-space form and time-discrete states which are the result of the event-dynamic superimposed structure. FMU use standardized variables for communications being the time, continuous state, time derivative of the state, parameters, inputs and outputs, discrete state and event indicators. The solver communicates the time and the continuous state, whilst getting back the state time derivative, the discrete state and the event indicator, whilst the input and output provide interfaces to other FMUs.
The FMI/FMU approach is providing a standard framework for the construction of new units. The use of the state-space notation as a core ensures a certain degree of universality which further increases portability.

2.5 Discussion

At the beginning it was the exchange of data, with physical properties and experimental data in the center. It extended to wrapping legacy-coded models and generically coupling tasks that solve models in a particular context and domain and finally one sees efforts that focus on the models itself. The subject is clearly developing but it is still far from being resolved. The method of coupling depends on the computing environment, processors, multi-tasking, parallel computing and recently cloud computing has been added. On this low level the message passing standards are bringing some generic order into the framework that do have the potential to cover the need for the different computing environments and are simple enough to be implemented in the respective application domains. Most computing environments and languages will support task control and communications can be made working across networks of different types.
There is also no lack of motivation, though commercial interests do act as breaks since the target of these efforts is integration, whilst commercial interest is to capture.
So what else is then the problem? The disappearance of the pdXi gives a strong indication, that it is mainly the structure of the communicated and stored information that must get into the focus. In order to find feasible solutions that find a reasonably wide acceptance, the representation of data and model must be abstracted to the degree that its use is easy to implement in different environments and it must become clearly layered so as to accommodate the different needs. The abstraction must include different levels enabling the interaction on different levels thereby matching different applications and level of use.

We must also accept that very many different computation environments are in daily use and that one must aim at a multi-language, multi-platform solution must be sought.

3. Structural Considerations for the Future

There are a number of issues to be considered in order to get a coupling technology accepted.

3.1 The Mathematical Problem

Numerical problems are defined by a set of equations and a set of instantiated variables, the givens and a set of variables for which the set of equations is to be solved using a method, which in terms may be parameterized and requires respective control information.
Thus the relevant pieces of information are:

- Variables and their documentation
- Index sets and their documentation
- Equations and their documentation
- Instantiated variables
- Unknown variables
- Method specification
- Method parameters

3.2 The Equations in a Mathematical Context

The equations and variables are defined in a particular mathematical context, such as ODE's, PDE's and algebraic equations. The variables that appear in time-derivatives, may be defined as the state variables, whilst those that are defined from outside of the equation set, may be either seen as inputs or parameters. The variables that are visible to the outside may be termed outputs.
The equations may also include distribution functions in the case of stochastic systems. In the case of event-discrete systems, difference equations may also be present.
The terminology used in mathematical system theory is perfectly suited to accommodate the various objects using the mathematical point of view.

3.3 The Equations in an Application Context

The equations are usually established in connection with a particular application. In the case the application is physical and a mechanistic representation is constructed, one talks about a mechanistic model and the dynamic equations are in the original form the conservation principles, thus (component) mass, energy and momentum. The conservation applies to capacities and is supplemented with algebraic equations that represent the coupling of the capacities through transport and internal dynamics representing the ongoing transposition of mass and energy from one form into another, thus reaction, phase changes and the like.
The model may be of finer granularity and comprise population balances in which the change of the population is part of the internal dynamics.
On paper this application context asks for a different set of names for the variables than the mathematical context. The application can thus be seen as a level above, re-grouping the variables and equations matching the applications context. On this level also the indices being used to abstract into sets or vectors are defined. This can be done with application-specific attributes.
Larger models may be assembled from a set of smaller ones. If the model components are being linked on this level and not on the mathematical level, the linking is done in the application context. For example two physical models may be coupled/linked/connected by a heat transfer or any other transfer of extensive quantity.

3.4 Extension to Hierarchical Context

The application context may well be hierarchical defining models and agglomeration on different scales, on different levels of detail. This then requires defining a hierarchical context in which the state-space models are defined on each level of detail. Transitions from one level to another may be required and in some case one may substitute parts of the model by surrogate models.

4. Considerations Regarding Implementation

The existing technologies of couplers and wrapper-type are often driven not only by the set problem of enabling interaction etc. but also by the technology available for the implementation. In fact one often gets

the impression that the available implementation technology has a greater impact on the chosen solution than the original problem formulation. This is largely due to the fact that the actual work of generating a coupler or similar environment is done by people that evolve from the technology implementation side and not from the application side. This is a reflection of the situation that many application fields have failed to educate the implementation technology, in this case what we broadly term programming or more finely computer engineering. Whilst chemical engineering held a leading position in computations in the 1960ties, it has failed to maintain this position and neglected it to the point where even the even the basic education is nearly non-existent. The subject has been left to the few specialists many of whom have in the meantime retired and correspondingly little is happening in this field.
Couplers are mostly coming out of the physics-driven community, with weather prediction being a major field, mechanical engineering in various applications is at least an equally strong driver and biology as well as medical applications are becoming increasingly visible and important.

Having said so, what direction could or should be taken. Paradigms shift all the time. In recent time, though, a shift towards a more descriptive technique can be observed. One of the main representative shifts go towards language-oriented programming, which pushes for the definition of application-specific languages that then are used for the representation of the problems in question. The objects represented in this context-sensitive language are then translated into different target languages that are suited for the lower-level application and active computational coupling operation.

Today we consider it feasible to define a context-dependent language for all the levels defined above and enable automatic code generation into any target language making the models and their associated information available for linking into the model solver domain. This would clearly contribute toward solving the multi-language problem, but also to a large extent remove some of the coupling operations as well as making it possible to do the same for multiple platforms.

5. Conclusions

From what we discussed it is clear that model exchange and data exchange has drawn the attention of the developers of various different fields. Currently several alternative solutions are available, though the foundations are very similar. The method of taking legacy coded models and wrapping them is favored because it has a minimal effect on the operations. Also it makes it possible to define the wrapper itself as a standard. Any implementation error is isolated to the wrapper and thus relatively easy to locate if one can trust the wrapped legacy code in the particular implementation, that is. The approach has the obvious disadvantage of increasing the communication overhead, which may render the final product infeasible to solve the problem in a reasonable or required time. Also the behavior of the underlying model is hidden behind layers of wrapping code.
Whilst Cape-Open is a typical product of a conservative compromise, LIME and FMI/DMU approaches raise the level of requirement but also provide significantly improved performance and ease of use. Overall it is felt that the developed concepts are too much influenced by the implementation technique and have too little descriptive power to capture a wide group of application domains. This applies to nearly all levels implying on the numerical computational level, the problem definition level and the application level. The creation of domain-specific languages and a layered approach would make the interpretation and the implementation more intuitive, more matching the context of the application in contrast to the execution. A language approach also enables verification using structural components like classes. The models will then be able to search and identify the relevant components in other units and call on them selectively. This avoids the passing of irrelevant information and leads to higher efficiency. Application-oriented structures make the models more logical and therefore user friendly. Instead of adding redundant layers to the model, the units are kept lean and effective.
The coupler concepts should be extended to naturally accommodate any numerical problem including integration, interpolation, matching, but also problems like optimization, to mention a definitely essential computational activity.

References

Blaha, M, 2010 Data modeling is important for SOA, Advances in Conceptual Modeling – Applications and Challenges, Lecture Notes in Comp Sci, Vol 6413, pp 225-264.

Larson, J W; 2010, Ten organising principles for coupling in multiphysics and multiscale models, ANSIAM J, Vol 48, C1090-C1111.

MPI-Forum, 2012, MPI: A message-Passing Interface Standard, V3.0, available from www.mpi-forum.org

Webpages:

CoLan, 2012, Webpage with documentation, www.colan.org (last visited 2013-02-04)

NSF, 1986, nsf.gov/awardsearch/showAward?AWD_ID=8514513 (last visited 2013-01-31)

MFI, 2013 Functional Mock-up Interface, www.fmi-standard.org (last visited 2013-02-07)

MpCCI, 2012, www.mpcci.de/multiphysics-engineering.html (last visited 2013-02-04)

OpenMP, 2013, openmp.org

OpenPalm, 2011, www.cerfacs.fr/globc/PALM_WEB (last visited 2013-02-04)

PDXI, 1995, http://pubs.acs.org/cen/hotarticles/cenear/950327/art12.html (last visited 2013-01-25)

TISC, 2012, www.tlk-thermo.com (last visited 2013-02-04)