

Model based soft-sensors based on OPC Unified Architecture

Paolo Greppi, Consultant, 3iP, Italy

Contents

Abstract.....	1
Context.....	2
OPC Unified Architecture.....	3
Soft sensors.....	5
The Soft Sensor as an OPC UA server.....	6
LIBPF.....	7
LIBPF architectures.....	9
Implementation.....	12
Comparison with Classic OPC.....	15
Conclusions.....	15

Abstract

Process modelling can be arranged as a domain-specific information model on top of OPC Unified Architecture, the new platform-independent standard for interoperability between enterprise information systems and industrial automation. With this technology model-based soft-sensors can be implemented as components in a Service Oriented Architecture, with the option to discover, set-up, configure and schedule the models from within the control system maintenance user interface.

In this work a prototype implementation of a first-principle soft sensor is presented, wrapped as an OPC UA Server. That soft-sensor calculates water and natural gas properties based respectively on the IAPWS-95 and GERG-2004 equations of state.

The thermodynamic calculations are implemented using LIBPF (LIBrary for Process Flowsheeting), a process flowsheeting and modelling tool arranged as a C++ library. The object-oriented design and portability of the LIBPF library fits perfectly with the philosophy of OPC UA, allowing for reduced development time and an extremely lean implementation.

The soft sensor can be deployed either in embedded devices or as a server daemon/service.

Context

The drive to react quickly to market changes is very strong in the power generation industry and it impacts all participants to the value chain.

To the supplier of sub-plants or packages it means they have to adapt their units to the changing specifications, and make them easier to deploy by adapting to the customers' industrial information technology infrastructure.

To the licensor and the engineering company, it means adapting the process scheme to the boundary conditions and the peculiarities of the destination site.

To the automation supplier it means that the process automation and information technology infrastructure should follow seamlessly the changes during the design and operation phases without requiring risky reconfiguration or manual adaptation of custom interfaces.

To the operating company it means they have to adjust their production mix to the demand by the minute, and handle feedstock changes. The production mix for cogeneration or trigeneration has many degrees of freedom which can be optimized, but day-night fluctuations in the electricity price on the grid pose the challenge also to large sized installations. While the feedstock is variable by nature in the case of biomasses, fossil fuels change quality continuously too.

The current information technology infrastructure and in particular industrial process automation is not exactly flexible, but the answer to the demand for faster reactivity to market changes lies in flexible, intelligent automation and open standards for vendor interoperability.

It is widely recognized¹ that a Service-Oriented Architecture can match the flexibility requirement, allowing the industry to use existing IT investments and achieve the flexibility required. SOA is an architectural approach to business process design and implementation in software that maps business processes to services, that are implemented as interoperable software modules.

1 Credle R., Akibola V., Karna V., Pannerselvam D., Pillai R. and Prasad S. "Discovering the Business Value Patterns of Chemical and Petroleum Integrated Information Framework" IBM redbooks 2009 ISBN 0738433136

Once the business process and the associated services are based on SOA, they are loosely coupled, reusable and easier to move outside and inside the boundaries of the company. Services such as diagnostic, training, maintenance and support which are simple to externalize will benefit from this kind of flexibility. Vice versa operating and licensing companies will want to keep in house the high value added services.

Once the migration to SOA is accomplished, a broader spectrum of applications such as mobile terminals, intelligent embedded devices, augmented reality and even cloud computing opens up, generating deep implications for the business models in the process simulation, optimization and training sectors, both real-time and off-line.

OPC Unified Architecture

For the process automation industry, the recently released OPC Unified Architecture (OPC UA) specification² builds on the successes of the established Classic OPC standard³ for interoperability between enterprise information systems and industrial automation.

With OPC UA data and functionalities can be made available as services in a vendor-independent fashion. The main characteristic of OCP UA are:

1. It exposes the semantics of the data in an object-oriented fashion;
2. It is platform-independent;
3. It is built around a service-oriented architecture (SOA).

The key element of OPC UA is the capability to expose the semantics of the data: a temperature is different from a pressure; a variable reserved for the storage of a temperature measured in the field is conceptually different from a variable used to store a calculated temperature; the temperature, pressure and composition for a given process fluid belong together; the capability to compute a process model is common to all different models.

The OPC UA standard does not define in detail the semantic for all possible domains and objects that could occur in a plant: to express these relationships, Variable and Objects types can be defined extending the built-in OPC UA types.

In this regard it is much more lightweight than STEP (Standard for the Exchange of Product Model Data⁴): rather than starting off with the ambitious objective of being a monolithic,

2 OPC Unified Architecture Specification 1.01 Parts 1-9, OPC Foundation 2009

3 OPC Data Access Specification, OPC Foundation 1996

4 ISO 10303-1:1994, Industrial automation systems and integration -- Product data representation and

integrated information model, it just provides a framework to define objects and object types, and connect them with relations.

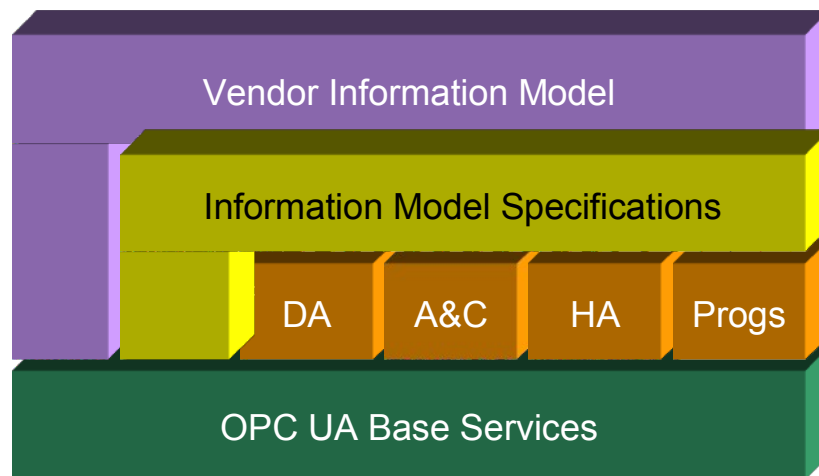


Figure 1 – OPC Unified Architecture Specification Layering

The figure 1, borrowed from a webinar of the OPC foundation⁵ shows how the information model specifications layer on top of the OPC information model, core and access type (Data Access, Alarms and Condition, Historical Access and Programs), to provide the domain-specific semantic information. Vendors may in turn further extend the domain-specific information model to differentiate and provide additional functionalities, interoperability being guaranteed by the common vendor-independent objects and relations.

Work is under way⁶ to standardize information models for physical device information, analyser devices, plant operation and maintenance, batch control, PLC programming.

Of course also soft sensors, process simulation, advanced control, model predictive control, plant-wide mass balance reconciliation or operator training could be encapsulated as OPC UA services. The benefits for the end-users would be striking:

- Platform flexibility: run on any operating system, run on wide spectrum of hardware from embedded devices (even the very same sensors and actuators in the field), mobile devices, workstations and servers;
- Interoperability and the possibility to assemble heterogeneous, multi-vendor solutions

exchange -- Part 1: Overview and fundamental principles, ISO TC 184/SC 4, 1994

5 OPC Unified Architecture Overview For Developers Architects Webinar PPT, <https://www.opcfoundation.org/DownloadFile.aspx?CM=3&RI=595&CU=120>

6 Wolfgang Mahnke, Stefan-Helmut Leitner and Matthias Damm "OPC Unified Architecture" Springer-Verlag 2009

thanks to the wide scope of connectivity;

- Increased flexibility and ease of use for configuration and maintenance: discovery of services, zero-configuration and plug-and-play integration with the rest of the IT infrastructure whenever possible, or reconfiguration after a plant or process change through OPC UA interfaces, using a third-party unified client maintenance interface;
- Business model flexibility thanks to the service-based technology;
- High Performance: low latency, low overheads and high data throughput;
- Reliability, security and redundancy;
- Easy migration plan for existing OPC products to the OPC UA technology.

Soft sensors

Soft sensors or inferentials can be considered as the simplest on-line application of models; they are therefore the best candidates to experiment with SOA and OPC UA.

Soft sensors are based on models that can replace real, physical sensors with virtual, calculated results; they are usually classified⁷ as either model-driven or data-driven.

Model-driven soft sensors are commonly based on first principle models based on the physical and chemical laws that underlie the physical processes, whereas data-driven soft sensors are based on some manipulation of the raw data such as Principle Component Analysis (PCA), Partial Least Squares (PLS) or Artificial Neural Networks (ANN).

Data-driven soft sensors are easier to set-up and have a lower computational cost, but are often non extrapolatable and suffer from drift. Model-driven soft sensors are not always applicable, require a time-consuming identification and model formulation step, and have typically a higher computational cost.

The simple process model which has been wrapped as an OPC UA Server in our prototype is a first-principle soft sensor that calculates a number of useful properties for a material stream composed of short-chain hydrocarbons:

- Lower / higher heating value and Wobbe-Index;
- Explodibility-related quantities such as LEL / UEL (lower and upper explosive limits) and LOC (limiting oxygen content);

⁷ P Kadlec, B Gabrys, S Strandt, Data-driven Soft Sensors in the process industry, *Computers and Chemical Engineering* 33 (2009) 795–814, doi:10.1016/j.compchemeng.2008.12.012

- Density, compression factor and temperature / pressure dew-point and bubble-point with an equation of state.

The typical application of this soft sensor is to provide inferential measurements when a process gas-chromatograph or other process analytic device is available and measures the composition of a process stream at regular intervals, such as in bio-gas and natural gas processing transport and distribution, power stations.

This kind of application in the natural gas sector is well-known and regulated by international standards⁸, with an ongoing evolution towards the application of more sophisticated equations of state, consequently the recent GERG-2004 equation was implemented in the prototype⁹. In the polymers and petrochemical sectors the interest is more in traditional cubic¹⁰ or modern statistical equations of state¹¹.

Due to the similarity of the implementation, the same soft sensor can be configured to compute water and water steam properties based on the IAPWS-95¹² equation of state.

To our knowledge this is the first working prototype of an OPC UA server for process simulation. In the following the approach used for the soft sensor implementation is described.

The Soft Sensor as an OPC UA server

There is no domain-specific information model for process simulation nor for soft sensors, therefore there is not yet a standard way to describe physical quantities, calculated variables, fluid-related objects.

In the implementation of the prototype certain unilateral choices were made:

- the EngineeringUnit property of the base variable type was used to distinguish

8 INTERNATIONAL STANDARD ISO 6976:1995 "Natural gas — Calculation of calorific values, density, relative density and Wobbe index from composition"; INTERNATIONAL STANDARD ISO 12213:2006 "Natural gas -- Calculation of compression factor -- Parts 1, 2 and 3"

9 O. Kunz, R. Klimeck, W. Wagner and M. Jaeschke, "The GERG-2004 Wide-Range Equation of State for Natural Gases and Other Mixtures", GERG TM15 2007

10 Ding-Yu Peng, and Donald B. Robinson, "A New Two-Constant Equation of State", Ind. Eng. Chem. Fundamen., 1976, 15 (1), 59-64 DOI: 10.1021/i160057a011

11 J. Gross and G. Sadowski, "Perturbed-chain SAFT: An equation of state based on a perturbation theory for chain molecules", Industrial & Engineering Chemistry Research, vol. 40, pp. 1244-1260, Feb 21 2001

12 Wagner, W., Pruß, A. The IAPWS formulation 1995 for the thermodynamic properties of ordinary water substance for general and scientific use. J. Phys. Chem. Ref. Data 31 (2002), 387-535.

different physical quantities

- the AccessLevel field was used to make a calculated variable read-only
- temperature, pressure and composition for a process fluid are grouped in a “stream” abstract object type;
- all concrete stream models are subtyped from an abstract modelBase object which has pure virtual methods Calculate, Reset and ResetErrors.

For the implementation of the OPC UA server the first and currently only commercially available high-level C++ software development kit (SDK)¹³ has been used.

The thermodynamic calculations and the empirical correlations are implemented using LIBPF (LIBrary for Process Flowsheeting), a process flowsheeting and modelling tool arranged as a C++ library¹⁴.

LIBPF

LIBPF (LIBrary for Process Flowsheeting) is a modelling tool for industrial, continuous processes. Its purpose is to make it possible to rapidly prototype and deploy solutions for training, process engineer support, on-line process diagnostic, and data reconciliation.

The necessary models (physical properties and unit operations) and tools (solvers, input/output, object persistency, communication interfaces) are made available in the form of a C++ library; the model developer writes simple C++ code and links it to the LIBPF library to create lightweight, special-purpose application to compute a specific process or process family.

The model user receives the model as a black-box executable with no access to the underlying equations and assumptions; she can play around with model inputs and configuration options but no change in the structure is possible.

The unique features of the LIBPF library are:

- Compatibility: it is written in standard C++¹⁵ resulting in high portability;
- Leanness: just about 30k LOC (Lines Of Code), thanks to extensive code reuse, resulting in a small footprint of the resulting calculation kernels (in the range 2 ÷ 4

13 C++ based OPC UA Server/Client SDK V1.0.0, Unified Automation GmbH 2009

14 P. Greppi, “LIBPF: A LIBRARY FOR PROCESS FLOWSHEETING IN C++” Proceeding of the International Mediterranean Modelling Multiconference 2006, pp. 435-440

15 ISO 14882:1998, “Programming Language C++” ISO/IEC 1998

MB);

- Completeness: over 250 object types (classes) make it possible to code the model directly in C++ effectively and reliably.

LIBPF maximises reuse of existing libraries and external tools, thereby reducing risks associated with reimplementation:

- STL (Standard Template Library) for data containers;
- OpenMP for multi-threading, to exploit performance enhancements in multi-core or multi-processor workstations;
- boost::graph for graph manipulation;
- AT&T GraphViz 2.17 for graph visualization;
- ADOL-C for Automatic Differentiation;
- GMM++ 3.05 (Generic Template Matrix C++ Library) for vector and matrix manipulation;
- CSparse: a Concise Sparse Matrix package;
- SUNDIALS for Differential and Differential-Algebraic equation solving.

LIBPF natively supports the following modelling features:

- Problem Resolution:
 - Static and dynamic processes;
 - Sequential resolution of flowsheets with material and heat recycle (direct substitution or accelerated direct substitution);
 - Process specifications (feed-back);
 - Sparse analytical derivatives;
 - Simultaneous solution of flowsheets, sparse linear solver;
- Physico-chemical properties:
 - Ideal properties (Raoult or Henry law and ideal gas law)
 - NRTL activity coefficient model
 - Basic cubic equations of state (Peng-Robinson and Soave-Redlich -Kwong);
 - Statistical equation of state PC-SAFT
 - Pure water and water steam equation of state IAPWS-95
 - Natural gas mixtures of up to 18 components with the GERG-2004 special-

purpose equation of state

- Gas phase, liquid phase or vapour-liquid phase processes.
- Unlimited number of solid phases;
- Built-in concentrated parameters unit operations models:
 - Stream mixer, 2 or more inlets;
 - Stream splitter (tee), 2 or more outlets;
 - Spawn (duplicates the inlet);
 - Fixed-yield separator, 2 or three outlet streams;
 - Optionally reactive (fixed yield stoichiometric or equilibrium) flash with phase equilibrium;
 - Isentropic compressor/expander;
 - Counter-current non-reactive adiabatic HTU/NTU absorber/stripper;
 - Reactive multi-stream heat exchanger;
 - Fuel cell with 2 or more streams, each supporting multiple equilibrium or fixed conversion reactions; one or more electrochemical reactions supported;
 - Flowsheet-in-Flowsheet;
- Building blocks for distributed parameters unit operations models:
 - Generic multi-stage 1-D unit, with co-/counter- current flow pattern;
 - Generic multi-stage 2-D unit, with spiral, co-, cross- or counter-flow pattern.

More complex unit operations (distributed parameter fuel cell, plug-flow reactor, zone heat exchanger, distillation column etc.) are obtained as combination of the generic multi-stage 1-D and 2-D unit with the concentrated parameter models. For example a 2-D distributed parameter planar fuel cell, with anode/cathode cross-flow can be assembled from a Generic multi-stage 2-D unit with an array of fuel cell concentrated parameters unit operations models.

LIBPF architectures

The LIBPF library and products allow to compose a variety of different architectures to suit the requirements of the application. Four basic configurations can be envisaged:

1. Desktop;
2. Enterprise
3. On-line;
4. Embedded.

The architecture for a **desktop application** is composed of three components:

1. User interface (LIBPF UI);
2. Calculation kernel: console application without any user interaction developed in C++ using the LIBPF library;
3. RDB (Relational Data-Base) for data storage / retrieval, via ODBC 3.0 driver.

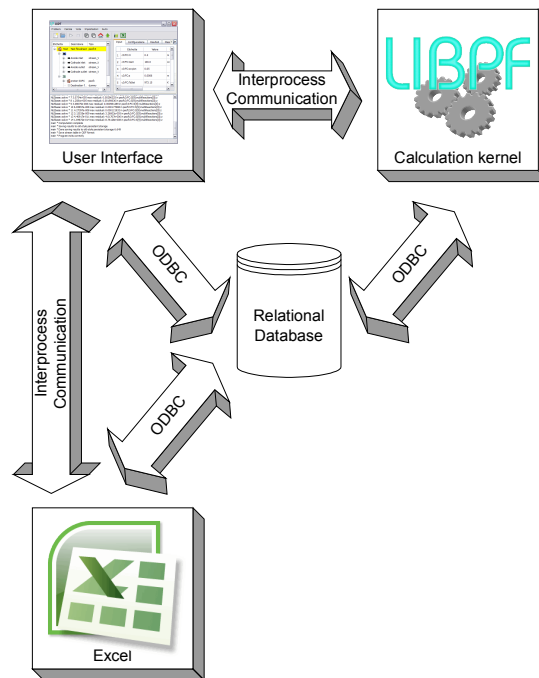


Figure 2 – LIBPF standalone architecture

The work-flow is: user interacts with the UI to initially instantiate an object of a give type (i.e. a process model with a given structure). User interacts further with the UI to change the specification; UI writes the new specification to the database via ODBC; user launches the calculation kernel to perform the computation and controls it via interprocess communication; UI interfaces to the database to get and display results.

The architecture for an **enterprise application** is composed of several components:

1. A number of user interface / calculation kernel pairs running locally and concurrently on the desktop of each user;
2. One centralized RDB (Relational Data-Base) to store models and data for all users and makes it possible to exchange data;
3. 3rd party applications can interact with the RDB to integrate with the other information infrastructure.

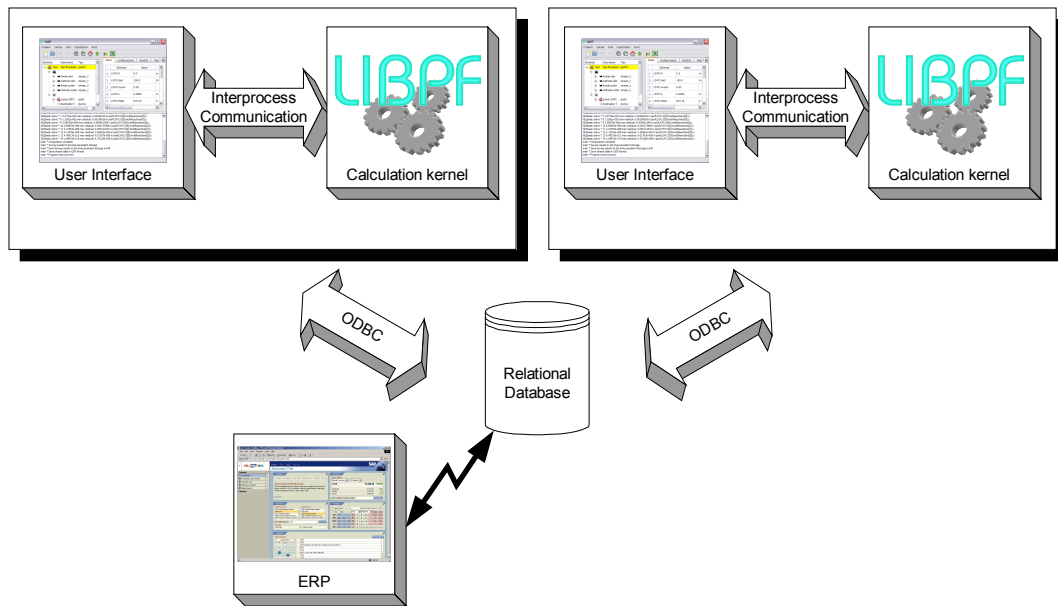


Figure 3 – LIBPF enterprise architecture

LIBPF can be used to develop the process-specific part of an on-line application. The calculation kernel is portable and lightweight so that it can be directly installed on the low-cost embedded hardware (such as PLC or SCADA), in a stripped-down **embedded configuration** consisting only of calculation kernel and RDB (Relational Data-Base):

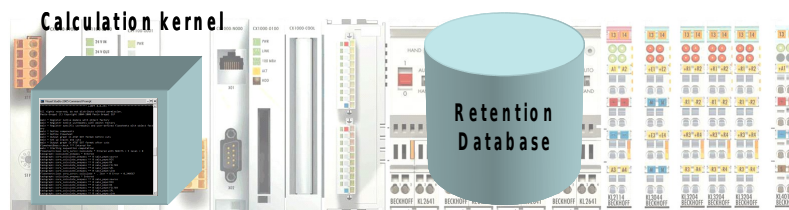


Figure 4 – LIBPF embedded architecture

The object-oriented design and portability of the LIBPF library fit with the philosophy of OPC UA. This reduced the development time and resulted in an extremely lean implementation: coding the LIBPF / OPC UA bridge required a mere 5000 lines of C++ code.

Finally, in another type of scenario the **on-line application** can be integrated in a more structured infrastructure, where a Classic OPC server or a client/server OPC UA is present; in this case the on-line architecture



Figure 5 – LIBPF on-line architecture

Implementation

The LIBPF architecture suitable for the OPC UA server prototype is the Figure 5 – LIBPF on-line architecture, figure 5 above. Furthermore the following key LIBPF functionalities were used for the prototype:

- units of measurements: all quantities in LIBPF carry a unit field which was used to populate the EngineeringUnit property of the corresponding OPC UA variables
- reflection: the ability to programmatically inspect and use types at run-time)
- dynamic type manipulation via the object factory without reference to Windows-specific object broker mechanisms such as COM or .NET
- object persistency.

Both the OPC UA SDK as well as the LIBPF library support both Windows and Linux, so the prototype server was successfully tested in both environments. If desired the soft sensor could even be integrated directly into the automation device, whatever real-time operating system it is running, thanks to the portability of both libraries.

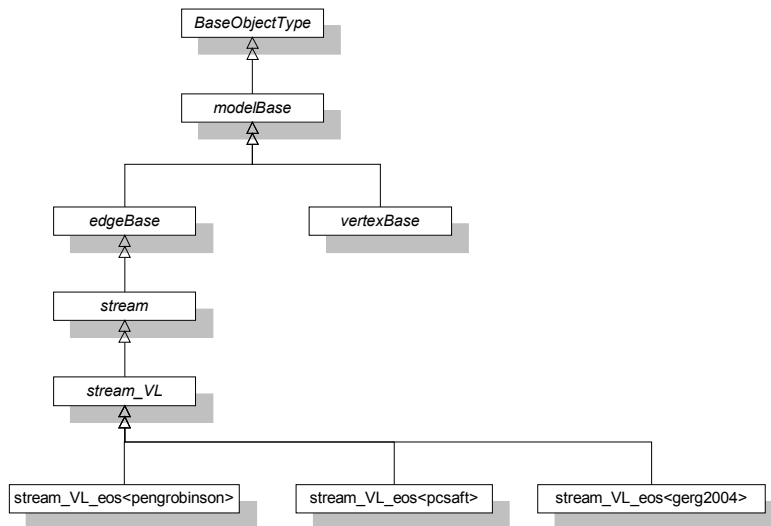


Figure 6 – OPC UA types from the implemented process simulation information model

An exemplificative subset of the implemented type hierarchy, expressed in the OPC UA (UML-derived) graphical notation is shown in figure 6. Shaded boxes indicate object types; object type names in italic indicate that the respective objects are abstract i.e. they express an incomplete set of characteristics, and can not therefore be instantiated but only contribute to the definition of a more complex type. The double-arrows connection indicate the types are in a HasSubtype relation.

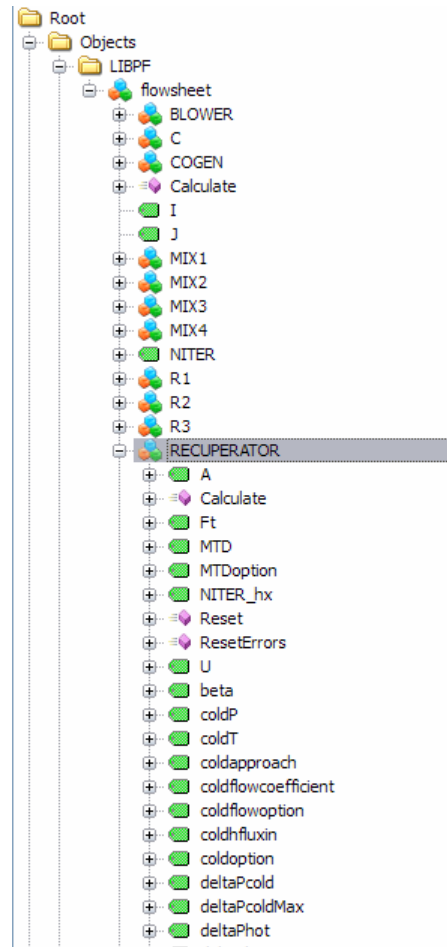


Figure 7 – Hierarchy of LIBPF objects as seen from UaExpert OPC UA client by Unified Automation GmbH

The instantiated objects exist in some form in the memory of the OPC UA server, as "live", instantiated objects of certain types. OPC UA clients can browse and access the objects, and execute methods on them; see figure 7 for a screenshot from an OPC UA client browsing the hierarchy of LIBPF objects for a more complex model representing a flowsheet. Either the server has a fixed set of live object instances, determined at compile-time, or it can expose a node creation service (note: this functionality is not yet implemented in our prototype). In this latter case clients can browse the hierarchy of available types known to the server, and request that one of them should be instantiated at run-time; this scenario makes sense for a simulation information model, where the types match virtual, non-physical entities, or during sensor/controller/actuator configuration for an hardware-related information model.

If the server has to serve node creation service requests, and the clients require that server-side objects "survive" a restart of the server process, the server needs a facility to persist the live instances. For this purpose LIBPF offers a built-in persistency mechanism to an external

relational database.

The persistency mechanism is also useful if the server exposes a large number of live instances, which are infrequently accessed: with persistency unused objects can be cached to disk reducing the consumption of resources.

Comparison with Classic OPC

A similar soft sensor had been previously implemented with a Classic OPC DA interface. The main difference with the new OPC UA server prototype is that the old-style soft sensor was an OPC client: the OPC tags required to store the results of the calculations had to be manually created on the Classic OPC server of the DCS; next the tag names had to be manually configured with the soft sensor using an XML file. Any change to the OPC server configuration would require a manual update of the client configuration; this is where the limited flexibility of a non-service oriented architecture becomes evident.

Another difference is that the Classic OPC client contained a bare-bone state machine to schedule the repeated execution of the calculations, but this state machine was opaque and again only configurable with a custom interface (in that case, values stored in the operating system registry). The orthodox way to achieve this in OPC UA (although not yet implemented in our prototype) would be to use the State Machine information model and the Program abstraction provided by the specification.

In the resulting OPC UA soft sensor there would be then no need for configuration files, registry settings or a maintenance user interface: the configuration could be performed using any OPC UA client, requiring no client-side configuration thanks to the discover capabilities.

The preferred use case would be to have a higher level OPC UA server, i.e. the one provided by the DCS vendor, controlling the soft sensor according to the “chained server” pattern. The elected OPC UA client to perform the set-up in this case would be the generic control system maintenance user interface itself.

Conclusions

We have presented a first-of-a-kind soft sensor prototype implemented as an OPC UA server. To fully exploit the OPC UA promise for portability, flexibility and performance the modelling calculation kernel used to implement the process simulation information model should be modern, designed around an object-oriented paradigm and portable; the C++

LIBrary for Process Flowsheeting satisfies these requirements and allowed fast prototyping of the application.

The soft sensor calculates certain properties such as heating value, explosibility, density, compression factor and dew-/bubble-points for hydrocarbon mixtures. The latter either with industry-standard cubic equations of state, or with more sophisticated and accurate equations of state for specific mixtures (GERG-2004) or finally with the PC-SAFT statistical equation of state. To make the prototype complete would require implementing the node creation interface, the State Machine information model and the Program OPC UA interface; in this way it would be possible to completely steer the soft sensor using OPC UA for set-up and configuration as well as for communication at run-time.

The prototype is a simple example of a entire new array of applications of process models to mobile terminals, intelligent embedded devices, augmented reality and even cloud computing. OPC UA, if accepted as an industry-wide standard, is a potentially revolutionary technology that could reshape the automation industry and the business models in the "process simulation and optimization" and in the "real-time process optimisation and training" sectors.