1st International Conference "From Scientific Computing to Computational Engineering"
1st IC-SCCE
Athens, 8-10 September, 2004

TOP-ENERGY – COMPUTATIONAL SUPPORT FOR ENERGY SYSTEM ENGINEERING PROCESSES

Eckardt Augenstein*, Gregor Wrobel†, Isabel Kuperjans* and Matthias Plessow†

* Chair of Technical Thermodynamics (LTT),
Aachen University,
52056 Aachen, Germany.
E-mail: top-energy@ltt.rwth-aachen.de, web page: http://www.ltt.rwth-aachen.de

† R&D Department Graph-based Engineering Systems,
Society for the Promotion of Applied Informatics (GFaI),
12489 Berlin, Germany.
E-mail: {wrobel, plessow}@gfai.de, web page: http://www.gfai.de

Keywords: Rational Energy Use, Energy System Simulation, Engineering Framework, Object Model, XML

Abstract: *Rational energy use in industry decreases costs, environmental pollution and the depletion of resources. Since energy costs are often of the same magnitude as the profit, reduced energy use considerably affects the company results. Yet, systematic, profound analysis and optimization of energetic processes are seldom conducted. In order to reduce the effort and increase the quality of energy system analysis, a computer aided method is being developed in a research project called TOP-Energy (Toolkit for Optimization of Industrial Energy Systems).*

The main aim of TOP-Energy is to support energy consultants in analyzing and optimizing industrial energy supply systems by providing modules for documentation, simulation and evaluation of energy systems with respect to energetic, economic and environmental aspects. To ensure an integrated workflow, TOP-Energy reflects a business process model derived from the German guideline, VDI 3922.

TOP-Energy consists of two major parts, a unifying framework and a set of modules: The former supplies the services of a modern GUI-application such as module-sensitive dialogs and presentations, flow sheet editing and report generation. It includes a sophisticated, abstract data model and operates as a software bus for modules allowing them to integrate their own XML-based data. The latter are technical modules, addressing particular aspects of energy system analysis such as economic analysis or system simulation.

An outstanding feature of TOP-Energy is the supply system simulator, called eSim, which is tailored to the consultant's needs. It evaluates energy supply concepts by means of historical demand data. In contrast to other simulator packages, eSim supersedes the costly modeling of control systems at an early stage of concept development by automatic optimization of component deployment.

1 INTRODUCTION

Today, energy savings and efficient use of energy are important means of sustainable development. Particularly in industry, an appreciable amount of primary energy can be saved by enhancing the existing energy supply and transformation systems. Apart from the possibility of avoiding energy losses due to deteriorated equipment and insufficient system control, a considerable amount of primary energy can be saved by designing optimized energy systems on the basis of techniques such as cogeneration, heat pumps and heat integration. For cogeneration, for example, this has been surveyed in [10]. Unfortunately, methods and supporting tools to find such potentials, for example exergy analysis and system simulation, are well known in science but rarely used in practice.

Since the costs of energy are frequently of the same magnitude as the company's profits, reducing energy costs can lead to an appreciable effect on the profitability of the company. On the other hand, any investment in the energy system of a company competes against investments in new products or production equipment. It has to be profitable over a short payback period, even though it is a long term investment. Furthermore, the high costs of expert advice with the risk of uncertain results are usually avoided. This leads to a situation in which significant potentials for saving primary energy and money are not employed.

For these reasons, it is desirable to take steps to reduce the efforts and costs of an analysis and subsequent improvement of industrial energy systems. The use of computer aided methods seem favorable for this task, as an integrated tool can speed-up work and offer effective methods for system analysis and optimization. In addition, aspects usually neglected in decision processes, like environmental protection can easily be incorporated by an automated evaluation of appropriate coefficients.

These considerations lead to the development of a new modular designed software called TOP-Energy (Toolkit for Optimization of Industrial Energy Systems), which is developed by the cooperation of several partners from universities and industry.

2 OBJECTIVES AND REQUIREMENTS

The main objective of TOP-Energy is to provide an application that supports the process of energy system optimization for small- and middle-sized industrial sites. This work is usually done by smaller energy engineering companies that can not afford expensive custom developments. Contrary to numerous computational tools focusing on a specific technical problem, TOP-Energy integrates all steps of the typical workflow in one application, supplying a seamless flow of information from the first step of a project to the last. Apart from the support of computational methods like system simulation, special attention was paid to combining pure scientific methods with services such as comfortable user I/O and report generation. It is expected that this approach enhances working results by allocating appropriate computational methods, as well as reducing routine work by the automation of standard calculations and integrated project management.

The basic requirements implied by this approach are as follows:

(1) The application must have a graphical user interface with the look and feel of contemporary applications. Modules containing scientific computation must be easily integrated including an appropriate user interface.

(2) Although a standard workflow exists throughout the branch, few consulting projects are carried out under ideal circumstances. Therefore, the workflow implemented in the application must be flexible in terms of starting and completing a project at arbitrary points in the workflow.

(3) While a large part of every energy-consulting project is covered by the standard workflow, a certain steps are specific to the individual project aims and industry branch. Therefore, one important requirement is the extensibility and adaptability of the application to the individual needs of the consultant engineer.

(4) Unproblematic data exchange between different installations and with third party applications is vital to the usability of the application.

3 BASIC APPROACH

The basic approach taken for the design of TOP-Energy is outlined by the following basic concepts incorporated into the application:

(1) A business process model describing a standard workflow for industrial energy system analysis and enhancement: This model was derived from the German guideline, VDI 3922 (energy advisory for industry and trade) [14]. It describes a typical process for problem solution and is divided into three phases: problem analysis, solution development and decision. In accordance with this guideline a set of modules is defined each covering a major step in the workflow: A preliminary analysis module serves to give an overview of the energetic and energy-economic situation of the examined site. This includes the analysis of the structures of energy requirements and energy supply tariffs, the documentation of the main energy conversions and usages as well as the determination of the company’s objectives in terms of costs, technical and environmental aspects. This is followed by a detailed analysis where the goal is to document the energy supply system and identify its weak points in order to develop measures which enhance rational energy use. In the solution development phase, a concept for an optimized system is found. For this purpose, TOP-Energy offers a system simulator designed for fast composition and simulation of energy supply system models. Typically, the engineer first runs the model of the actual system in the simulator to verify the input data and to obtain a reference. Then, different modifications are considered and the variants are successively developed, each optimized according to one or more objectives of the project. The results of the simulation contain various energetic, economic and ecological coefficients. Finally, to assist decision making, the developed variants are evaluated comparatively using one as a reference. For economical aspects, this is performed by an economy module capable of computing a variety of key figures directly from the simulation results.

(2) A system boundary model describing the relations between the different sub-systems involved in the industrial energy conversion and use: While VDI 3922 describes the engineer’s workflow very well, it lacks a precise definition for system boundaries, especially the distinction between the energy supply system and energy utilizing systems. As this is vital for the interpretation of balance results and system coefficients, an appropriate definition is introduced in TOP-Energy. Basically, it distinguishes between four subsystems: First, the public energy market which represents the economic boundary conditions, supplying various forms of end energy to the examined site. Second, an environment based on the definition of a thermodynamical environment capable of exchanging energy and matter with the remaining two subsystems covering the

examined site itself: The energy supply system and the systems of energy end users (usually the production site). The former converts the available forms of end energy into forms required by the end users. The latter employs the delivered energy in all kinds of technical processes, possibly rejecting waste heat or production residues to the energy supply system. While the current development primarily concentrates on the analysis of the supply system, it is planned to extend the focus of examination to the energy utilizing systems in future.

(3) A separation of common application services on the one hand, and the core functionality addressing specific technical problems or tasks on the other: Analyzing the basic use cases of the application, it became apparent that every technical module requires certain functionalities that are non-specific to the technical problem it addresses: Basically, these are the persistent data storage and data exchange, user interface functionality like I/O-dialogs and flow sheeting, managing projects and report generation. In case these services are outsourced to a generic framework which exports appropriate interfaces, the development of a module is reduced to the implementation of its technical contents. On the one hand, the availability of such a framework simplifies the transfer of a scientific program code to an engineering application tremendously. On the other hand, cooperation on the level of technical contents between framework, modules and - as is shown later - component models, makes high demands on interface specifications and the underlying object model (see Section 6).

(4) A user-role model distinguishing between different user perspectives: Claiming extensibility and adaptability, the need arises to define the level of user-friendliness the application offers to accomplish accordant changes to the system configuration. As extensions or adaptations of the contents, e.g. adaptations to certain industry branches, should be made by energy professionals without the help of software engineers, sophisticated programming skills are not expected. Therefore, besides module programmers being familiar with software development, two basic user roles are introduced into the system, the *consultant* and the *system operator*. The consultant represents the engineer focusing on the analysis and optimization of energy systems. They make use of the technical functionality of the application to gain profits for their work without altering the current system configuration. In contrast, the system operator has deeper knowledge of the application and is capable of adapting the system configuration by supplying *component templates* to the consultant (see below).

(5) A unifying object model capable of mapping the component and network based structures of the engineer's real world: As the framework supplies services for persistent data storage and data manipulation on the user interface to the modules, sophisticated interface specifications are needed. The foundation for these definitions is an object model describing the structure and semantics of data to be exchanged. Its central element – the component – is used to map real-world units, as well as represent the usage of a module within the project tree. The model is described in more detail in Section 5.1.

(6) Closely related to the object model, a database supporting the development of technical data structures: Using well defined technical terms when requesting user input or presenting results is vital to the usefulness of the application. Furthermore, developing a set of cooperating modules and component models requires a method to guarantee identical semantic interpretation of data by all communicating elements. This is achieved by setting up a global dictionary of technical terms and making reference to a so-defined term becomes a part of every interface value declaration. For details, see Section 5.2.

(7) A template-document approach addressing the instantiation of data objects in the application: In a simple, non-generic application, data objects are directly represented by the objects instanciated from the implementing classes. The classes define the structure of the data objects as well as their semantics, the data objects can neither exist nor their contents be interpreted without these classes. Using a generic data model structure, individual semantics and appearance of a data object are not defined by the implementing classes anymore, but by some kind of external template description. Instanciation of a data object from such a template can be done in two different ways: Using a class-object approach, the data object's relation to its template is the same as it is between the data object and its implementing class in non-generic implementations. Choosing a template-document approach, the data object inherits all information from its template and therefore, can exist and is interpreted independently from its template. Paying the price for redundant data, this approach reduces the complexity of the object model's implementation and increases flexibility and was therefore selected for TOP-Energy.

4 SCIENTIFIC COMPUTATIONS

4.1. System simulation

TOP-Energy was designed to incorporate various computational methods implemented as plug-in modules. The most prominent one to be part of the first release of TOP-Energy is *eSim*, a module for simulation of energy supply systems. The main objective of this module is to determine a set of economic, energy-related and environmental key figures of an energy supply infrastructure for a term of usually one year. The boundary conditions of a system model are demand profiles for the different forms of energy as well as information about the environment.

Energy system simulation is well known in science today. Various different solutions are available [5] although only few meet the usability requirements of engineering applications. Depending on the specific objectives of a simulation system, different mathematical approaches are used. The most sophisticated models base on differential equations (either DAE or ODE), being able to forecast the dynamic behavior of an energy supply system [15]. This, of course, requires a detailed modeling of the feedback control systems. Apart from that, detailed design data is required to calculate a machine’s time constants. In a more simple approach, the inertia of the system components is neglected due to the fact that the time resolution of boundary conditions usually is not better than one hour [1]. These quasi-static models result in algebraic systems requiring less detailed design parameters of the system components. However, the problem remains that control rules must be supplied by the user.

An approach that supersedes the specification of any control rules is to determine the system state by means of operation optimization. These systems, usually based on mixed integer linear optimization (MILP), are frequently used to determine optimal power plant schedules [11]. However, this approach is capable of evaluating the design of energy supply systems too [2]. One drawback of these optimizing systems is the fact, that, due to the limitations of the underlying MILP, their models regard media temperatures to be state independent, i.e. constant, or must be estimated before the linear problem is solved.

As eSim is thought to be used in an early stage of planning and modeling a system should not be too time consuming for the user, the operating optimization approach was chosen. To allow media temperatures to be variable, mass equations must be solved in addition to energy balances. As existing approaches to incorporate mass and energy equations into one MILP system [6],[8] did not meet the requirements in terms of performance and accuracy, eSim uses two separated systems, coordinated by an evolutionary optimization algorithm.

4.2. Scheme layout

Schematic representations of net-like structures are important for the support of human thinking and decision making. Furthermore, schemes are substantial parts of technical documentations. Naturally, the designing of such extensive structures is time-consuming and error-prone.

To solve this problem, graphic editors that provides creating appropriate layouts of networks by graphic interactions, can be used. The variation possibilities of the associated layout problems are diversified over the different application areas. Several layout methods are collected within a toolbox that is contained in a base framework.

TOP-Energy includes a flow sheet editor for the modeling of energy supply systems. Because of the marginal spatial comprehensiveness of these systems, a special placement procedure is not necessary. The included local alignment operations are right sufficient. However, strong internal linking structures require an appropriate routing procedure. Therefore, a method for drawing connections in such energy systems in an orthogonal style was developed. This method consists of three parts: pattern routing for two point connections, autonomous bus routing and post processing for bus routing. The pattern routing draws two-point connections in a sophisticated way by using a set of defined base patterns. Each of these patterns satisfies a certain routing situation. The method solves various situations by transforming to the base patterns. The bus routing handles hyper-edges and embeds them as sub-optimal Steiner-Trees. While the autonomous bus routing disregards collisions, the post processing optimized the results by regarding collisions. This routing method is part of a collection of placement and routing methods that is embedded in the system architecture (see section 6).

5 APPLICATION OBJECT MODEL

Whilst the business process model and the system boundary concept reflect the technical requirements of energy consulting projects, the separation of common application services and technical functionality, as well as the role-and-template based approach are consequences of the demand for extensibility and adaptability. All these different concepts are linked by the object model defining structure and semantics of any data object handled by the system. For brevity, this model is called TOM (TOP-Energy Object Model).

Tracking the lifeline of an arbitrary data object, e.g. a chiller component model, it starts with the creation of its template. Running the application in the system operator mode, a set of editors support the manipulation of the different component properties such as flow sheet symbol, network connection points (*pins*) and technical data structures. Furthermore, the system operator can add initial data, e.g. the simulator’s model description. The ready-made component template can then be used by the consultant by dragging it from the template library onto a module symbol or into a flow sheet and thus instanciating a component. The chiller model component is now a part of a flow sheet and after editing the technical data via the component’s forms, the consultant passes the flow sheet component to a module, e.g. the simulator, by selecting a menu command. In order to be handled by the module, the component needs to meet some requirements. E.g. the simulator will expect to find the component’s model description as a part of the technical data. The compliance of a component to a certain module therefore must be signaled by special flags set by the system operator during template creation. The module usually will manipulate some of the component's data and the consultant can view the calculation results via appropriate forms supplied by the component, possibly generate reports from, or simply store them together with other project data.

5.1. The TOP-Energy object model (TOM)

As one can see from this example, components are the central element of the object model. Furthermore, they are not restricted to mapping real world objects to the data space, but may represent any configurable element of the application as long as it forms a complete and autonomous unit. Therefore, different sub-classes of components are introduced:

- All information for a consulting project is gathered into one component tree. To give a certain structure to this tree, *Tree Components* are used. The most prominent tree component is the project component representing the root node of the project tree.
- *Library Components* are used to represent component collections. Each project has exactly two library components, one for component templates and one for components instances used in the project.
- *Module Components* map the use of a module to the project tree. They hold module specific parameters and results as well as special references to natural components associated with them (see below).
- *Natural Components* represent real world objects. Templates to instanciate such components from are held in the project's template library, the components themselves, no matter in which module they are used, reside in the component library.
- *Scheme Components* are very similar to natural components. The major difference between them is the consultant can manipulate the inner structure of a scheme (e.g. a flow sheet) whereas for natural components, this is reserved for the system operator.

Any of these components can aggregate a set of different elements in order to store information about technical data, GUI appearance, network structures and report generation. First of all, components can contain other components and thus are able to build tree structures or collections. However, depending on the component sub-class, the aggregation of components is restricted. In order to map network structures, components can contain connection points (*pins*) that can be connected to internal or external networks. This concept is identical to its underlying implementation, ELADO, which is described more detailed in Section 6.

Furthermore, components must be able to hold technical data in a way that modules can interpret it, even if they have no, or only partial knowledge about the specific data structure of the component. An appropriate solution to this problem is vital for the usability of the whole system as most natural components are expected to be compatible to about three or four different modules, even in the first release and therefore, the number of interdependencies is high. Another problem closely related concerns the interpretation of technical data by the user. In dealing with a vast variety of over 1000 terms from the fields of science, energy engineering and energy economics, a precise definition for each term and the exclusion of synonyms from the set of used data field names is crucial in avoiding misinterpretations by the user. The concept proposed here is based on tree structures of technical data, assembled during template creation, from a database of elementary definitions uniting an identifier, a technical term, its semantic definition and a data type. These elements are called *primitives* and are described in more detail in Section 5.2. In order to display this generically assembled data to the user, components may contain form descriptions, which are interpreted by the framework to generate I/O dialogs from elementary form snippets (*formlets*) corresponding to the primitive data types available.

Finally, special attention must be paid to the collaboration of natural components and modules: First of all, a natural component must carry information to which modules it is compatible, i.e. which modules the component template is prepared for. This is realized by a plug-socket mechanism, with the module component having a *slot* definition, and the module signaling its compatibility to this module by a corresponding *target* flag. In alignment with the target definition, the component guarantees to hold a module specific *primitive* tree. Second, by stating that a natural component maps exactly one physical object (e.g. a boiler) it can exist only once in each project, i.e. exactly one instance exists representing one physical object. However, it is possible to "use" this component in different variants or scenarios, generating an arbitrary number of component specific result sets. In order to map this 1:n relation between components and their result sets, modules using a component neither refer to a component directly nor own it but hold a *ComptontentUse* object, which can be thought of as a partially-synchronized copy of the component: While the global data, (typically design data), always refers to the original component, the local data, (typically result data and always module specific), is held in the *ComponentUse* object.

5.2. Primitives

As mentioned in Section 5.1, all technical data of a component is assembled by pre-defined atomic data elements called primitives. Basically, each primitive incorporates a technical term (such as “heat flux”), its precise definition, a data type from a set of pre-defined basic data types (such as *number* or *characteristic diagram*), some data-type specific template data (such as upper and lower boundaries for *numbers*) as well as the appropriate structures to store the actual contents. The primitives are created and assembled in an external application implemented in a LAMP[1]-environment and finally imported into the component templates. Using a central primitive database, all system operators who are designing component templates are able to fall back on the same set of data elements, well defined in terms of meaning and structure.

As the number of primitives defined will easily exceed 1000 at the end of the project, the primitive collection is structured by means of object-oriented principles, modeling the lexical relations of the terms introduced in order to preserve usability: First of all, name spaces were introduced to categorize primitives by meaning and so to separate the different meanings of homonyms. The lexical relation of hyperonyms and their corresponding hyponyms (such as “energy” to “heat” and “electric energy”) are mapped by an inheritance mechanism allowing the override of single template data fields but preserving the data type. Finally, aggregation can be modeled by a special primitive type called *bundle*, allowing the creation of tree structures. Creating bundles of primitives in most cases goes along with placing them into a certain context (e.g. “temperature” becomes an outlet temperature by inserting “temperature” into the bundle “outlet state”).

6 SYSTEM ARCHITECTURE

This section describes the TOP-Energy system architecture, including different software sub-systems. The basic structure is shown as the UML 2.0 composite structure diagram in Figure 1. The light gray parts represent universal sub-systems, whose common functionality is encapsulated by an interface implementation of TOM (dark gray in Figure 1). This sub-system provides methods for proper integration of application components (modules). Modules provide the technical functionality, such as energy system simulation.

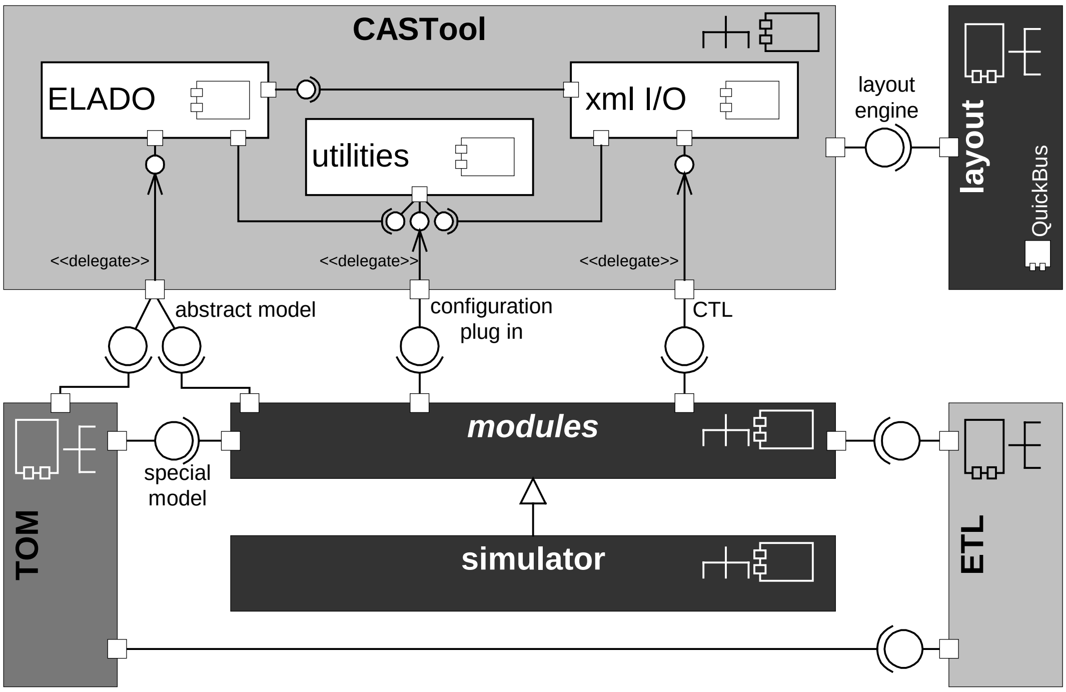

Figure 1. Architecture of TOP-Energy.

The core components are part of a domain framework called CASTool (*Computer Aided Schematics Toolbox*). The development of CASTool started in 1996 at the GFaI [12] and is now the basis for product line development in the field of graph-based engineering systems. These systems have in common their underlying model structure which is an extended mathematical graph, a so-called network. Such networks consist of network-components whose relations to each other are described by connections (nets). In previous research projects, various applications were developed such as for switch cabinet construction [13] and process control engineering [4].

A central component of CASTool is the data model implementation called ELADO (*Extended Layout Data*

[1] LAMP: Linux-Apache-MySQL-PHP, typical set-up of open source software for the implementation of web applications.

Model). ELADO is the fundamental domain model and core of CASTool. It consists of three parts, a domain or structure model, an application model and a representation model. The structure model of ELADO represents networks. While the common mathematical graph model includes vertices and edges, networks provide components, nets and pins. In extension to vertices in graph theory, components possess a shape as well as a symbol for graphical representation and can contain sub-networks. Nets can be seen analogous to edges in graphs and stand for two-point connections as well as hyperedges. Pins are connection points located on a defined place on a component's shape. The domain model of ELADO represents the domain specific model structure. Therefore, each relevant network element is represented by a class and appropriate access methods in the implementation of ELADO. Furthermore, ELADO provides an abstract application model. This part of ELADO stores required application data, such as technical and business data, in a generic tree structure, guarantying the re-use in different applications. In TOP-Energy, the ELADO application model is used to map primitive trees as described in Section 5.2, and to store template information like form descriptions. The third part of ELADO, the representation model, consists of data structures to store further information such as appearances, positions and layout behaviors.

CASTool also contains input/output components to serialize data. One of these components implements an XML-based data import and export. The underlying XML schema is called CTL (*CASTool Markup Language*) [3] and is capable of mapping the essential ELADO elements.

Another collection of core classes, represented by the component *utilities* in Figure 1, contains several utility components such as base data structures as well as configuration and command [7] management utilities. Command management implies a runtime registration and a call management as well as a command queue processing for redo and undo. Therefore, the application can be extended with new functionality by implementation and registration of new commands. Additionally, many features of a specific application can be configured using configuration files. This applies to GUI features such as menu and tool bars, project tree interactions and layout behaviors, as well as to module registration.

An extension developed for the TOP-Energy project is the component-dependent generation of forms described in XML. By this, a generic and comfortable GUI for component-specific user I/O is provided. Several further components provide editors and other domain-specified functionality, e.g. the creation of component symbols and shapes as well as flow sheet editors.

Besides the CASTool, several application independent layout methods exist (see section 4.2). These layout components are integrated into the architecture by a layout engine (see Figure 1). This interface gives layout methods access to ELADO and integrates them into the GUI.

Another important sub-system of TOP-Energy is the *Engineering Type Library* (ETL). This class library, developed at the LTT, provides typical engineering primitive data types from simple types such as numbers (floating point numbers with unit and data state information) up to complex components like time series. Along with these types, many functions for data manipulation and algorithms such as generic unit conversion routines and interpolation methods are provided. Again, the ETL is an application-independent software component and is used as the basis in other research projects in the field of thermodynamics.

For an appropriate use of the abstract ELADO model, a bridge [7] component is implemented, representing the *TOP-Energy Object Model* (TOM). This component is not a model in the sense that it stores data, as all project data is stored in ELADO. The TOM component rather wraps ELADO, especially its abstract application model, and provides suitable access methods to project data. In particular, TOM serves as an interface with technical modules, delegating all data queries to ELADO. The results are returned as ETL objects, ready to use in the module code. Besides providing a comfortable and TOM-compliant access to project data, this module decouples the implementations of TOP-Energy modules and ELADO, guarantying a stable API.

Finally, to provide application functionality, the system architecture is completed by modules symbolized as an abstract sub-system in Figure 1. This is a placeholder for specific modules such as the preliminary analysis or the simulator. Modules are integrated into the system by the *plug in* and *configuration* interface. The *plug in* interface manages the registration of a module and therefore, the functions access, while the *configuration* interface incorporates a module into the GUI. As described above, a module obtains data via the TOM component. Furthermore, the CTL interface enables a file-based data exchange, allowing the incorporation of existing modules, stand-alone or grid applications. Altogether, this concept provides an appropriate embedding of system functionality, providing an enormous simplification regarding the transformation of scientific computation to engineering applications.

7 CONCLUSIONS AND PERSPECTIVES

The main aim of TOP-Energy was the development of a software system to support the analysis and optimization of energy supply systems. Claiming cheap integration of additional computational methods and user-specific adaptability, a generic framework incorporating technical modules was developed. Important underlying concepts are a role- and template-based approach, as well as a sophisticated object model. As a basis for implementation, the CASTool domain framework developed by the GFaI, was chosen, supplemented by the specialized engineering type library ETL. However, the implementation of such generic systems is complex and time-consuming and it turns out, that the transition from a scientific solution to a suitable engineering system with ergonomic and user-focused interfaces is an extensive step.

Further development of TOP-Energy will include a sophisticated version control for modules and component templates, as well as enhanced report generation. More information about TOP-Energy can be found on the project website at http://top-energy.ltt.rwth-aachen.de.

8 ACKNOWLEDGEMENTS

The authors would like to thank the following persons which are working in the TOP-Energy project team: Klaus Lucas, Stefan Herbergs, Jan Feller, LTT, Aachen; Michael Pocher, Oliver Berndt, GFaI, Berlin; Jörg Meyer, EUtech GbR, Aachen; Rainer Küffmeier, BFT GmbH, Aachen. The authors are also grateful to the Bundesministeriums für Wirtschaft und Technologie (BMWA) and the German Federation of Industrial Cooperative Research Associations "Otto von Guericke" (AIF) for the financial support of the project. The work was done from 01.01.2003 to 31.12.2004 under project code 92 ZBG / 1.

REFERENCES

[1] Augenstein, E., Kuperjans, I., Lucas, K.: “EUSEBIA Decision-Support-System for Technical, Economical and Ecological Design and Evaluation of Industrial Energy Systems”, in Tsatsaronis, G., Moran, M.J., Cziesla, F., Bruckner, T. (editors.): *Proceedings of ECOS 2002*, Berlin, 2002, pp. 446-453.

[2] Bruckner, T., *Dynamische Energie- und Emissionsoptimierung regionaler Energiesysteme*, Phd-Thesis, Institut für theoretische Physik, Universität Würzburg, 1997.

[3] CASTool Markup Language, http://www.gfai.de/xsd/ctl/CTL-Root.xsd.

[4] Ebert, R.-E., Goetze, B., Pleßow, M., Pocher, M., "Ein universelles System für die Projektierung von Prozessleitsystemen", A&D Kompendium 2002, publish industry Verlag GmbH, München, pp. 84-85.

[5] Elmegaard, B., Houbak, N., “Software for the simulation of Power Plant Processes, Part A and B” , in Tsatsaronis, G., Moran, M.J., Cziesla, F., Bruckner, T. (editors.): *Proceedings of ECOS 2002*, Berlin, 2002, pp. 742-772.

[6] Filter, S.: *Zur Modellgenauigkeit der mittelfristigen Einsatzoptimierung von Querverbundsystemen*. Phd-Thesis, RWTH, Aachen, 2001.

[7] Gamma, E., Helm, R., Johnson, R., Vlissides, J. (1994), *Design Patterns – Elements of Reusable Object-Oriented Software*, Reading, MA: Addison-Wesley Publishing Company.

[8] Hackländer, P. and J. Verstege: „Optimierung der Betriebsplanung von Gasversorgungsunternehmen.“ In: Energietechnik, VDI Gesellschaft (editor): *Fortschrittliche Energiewandlung und -anwendung*, Band 1594 der Reihe VDI-Berichte, pp. 605–614, 2001.

[9] Jochem, E., and Bradke, H., “Rationelle Energieverwendung in der Industrie; Erreichtes, bestehende Hemmnisse und Perspektiven”, Energiewirtschaftliche Tagesfragen Vol. 46, No. 8, 1996, pp. 478-482.

[10]Lucas, K., “On the thermodynamics of cogeneration”, Int. J. Therm. Sci. Vol. 39, 2000, pp. 1039-1046.

[11]Maubach, K.-D.: *Mittelfristige Einsatzoptimierung in Versorungssystemen mit Kraft-Wärme-Kopplung*. Phd-Thesis, Bergische Universität-Gesamthoschschule Wuppertal, 1994.

[12]Pleßow, M., Pocher, M., Schmid, M., (1996), "Konzipierung und Entwicklung einer Toolbox für Systeme zum automatischen Generieren von schematischen Darstellungen" *Proceedings of 2. GI-Workshop Entwurf und Dokumentation im rechnergestützten Facility-Management, Eisenach, Germany, 9.-11. October 1996, Eisenach (Wartburg)*, Vol. I, pp. 25-38.

[13]Pleßow, M., Wrobel, G., (2000), "Ein ganzheitlicher Ansatz zum automatisierten Entwurf von Schaltanlagen" *Proceedings of CAD 2000, Berlin, Germany, 2-3 March 2000*, Vol. I, pp. 263-282.

[14]VDI 3922, *Energy Consulting in Industry and Business*, Verein Deutscher Ingenieure, 1998.

[15]Wischhusen, S., B. Lüdemann and G. Schmitz: “Economical Analysis of Complex Heating an Cooling Systems with the Simulation Tool HKSim”. In: Fritzson, Peter (editor): *Proceedings of the third International Modelica Conference*, pp. 259–268, November 2003.