Framework for Simulation Integration

Radek Šindelář * **Petr Novák** *,**

* *Christian Doppler Laboratory for*
Software Engineering Integration for Flexible Automation Systems
Technische Universität Wien, A-1140 Vienna, Austria
(e-mail: {sindelar, novak}@ifs.tuwien.ac.at).
** *Department of Cybernetics*
Czech Technical University
Karlovo nám. 13, 121 35 Prague, Czech Republic

Abstract: This paper presents an idea of integration of simulations into the control system architecture. The benefits of Advanced Process Control (APC) has been clearly shown in the last years. Advanced control methods rely on quality models and simulations. These models cannot be developed and operated without access to online and historical data. Integration of simulators, data sources, optimizers and other tools is discussed in this paper. The whole problem is divided into two main phases and the first one is presented here. This phase is design oriented but it can be used for solving specific tasks. We illustrate the idea on the real world example.

Keywords: simulation, integration, process control

1. INTRODUCTION

Simulations and models of processes can be utilized in all phases of the control system life cycle. Simulations have been traditionally used by researchers as a tool for studying processes, solving problems in the real systems and for process automation design. But the progress in the last decade made process simulations more suitable for everyday engineering. Instead of researchers there are several groups of potential users exploiting modeling techniques in a very wide variety of reasons.

Despite of this there is no standard support of simulations and models in any phase of their life cycle. In the automation system two main phases have to be observed: design phase with the restricted access to plant resources and operation phase which is applied in the target plant environment. In this paper the concept of integration environment is proposed. The integration platform should not only technically connect tools with different nature but also allow the easy transition between all phases of simulation life cycle.

The remainder of this work is structured as follows: in the next section the motivation and general requirements are discussed. The section 3 is addressed to related works and projects on a simulation and integration problem. The section 4 introduces the concept and structure of environment for simulation integration. In the section 5 beside the current status the real project is described. The section 6 concludes and presents the future work.

2. MOTIVATION AND REQUIREMENTS

Integration means finding an architecture and general methodology for connecting heterogeneous parts which take part in the simulation life cycle. The problem of integration can be divided into two subparts - physical (technical) integration using service oriented interfaces and semantic integration. While the first subproblem can be regarded as the problem of finding the suitable integration platform, the second subproblem needs to involve such steps like data and object model for storing and handling with data, message transformation etc. The goal should not be to incorporate only one particular simulation environment or one group of simulators with similar object model and purposes but to prepare the environment which is able to deal with event based, dynamic or agent based simulators.

The project is focused on simulations and models of industrial processes. Their simulators and models can be used in a wide range of applications.

- model based control (advanced process control methods)
- operator training
- decision support
- estimation of unmeasured variables
- fault detection
- job planning
- operation optimization
- operation analysis
- automation design and testing

Some above mentioned tasks can successfully work only when simulations tightly cooperate with the SCADA system. SCADA stands for Supervisory Control And Data Acquisition. In the architecture of the industrial control systems often represents a subsystem responsible for system monitoring, data collection and process control. The SCADA system structure and its functionality differs vendor from vendor but it can be generally divided in the following subparts:

HMI - human machine interface shows data and process status to human operator and through it operator controls the process.
Databases and data servers - they are responsible for storing live data (actual values) and historical data.
DCS - distributed control system, all PLCs and other field devices, communication infrastructure are often regarded as a part of SCADA system.

In some industrial areas the differences between DCS and SCADA systems are very small and they are considered to be the same. In our approach DCS system can work as the stand alone system without HMI, databases etc. It means that DCS controls the process in the real time while the SCADA system supervises this control and coordinate it with other parts of the process.

It is natural that before a model goes to daily operation it must pass through initial phases of its life cycle. Basically the following phases of life cycle can be recognized for simulations

- design
  - preliminary analysis
  - step tests
  - filtering
- testing
  - off-line - simulation output
  - online - Simulation/SCADA system communication
  - HMI testing
- operation
  - operation modes
  - model updates

The main goal of the integration process is to provide an easy transition between the phases. It should also allow reusing simulation modules and integrate different simulation tools (in general heterogonous simulation tools - fuzzy rule based, dynamic simulation, agent based etc.). The current approaches to the simulation integration usually focus on the operation and do not involve all phases.

Models and simulations have been mainly used by scientists and researchers to answer their questions. In this communities advanced methods for dealing with data and models are not required because the skillful user is expected. When introducing models and simulation into everyday usage several issues arise. These issues could be divided into three groups

(1) Synchronization
(2) Data access
(3) User

2.1 Synchronization

When used for control two basic interoperability modes of simulations can be distinguished - SCADA-time operation and simulation-time operation. In the SCADA-time operation mode simulations and models run in parallel to the real controlled process and either aims (i) aims at the **estimation** of values that cannot be directly measured in the real system (e.g., soft sensors) or (ii) the selected data measured in the real system is compared with corresponding values provided by the simulated system in order to detect **an extraordinary situation**, possibly caused by a fault.

Very often the simulation-time operation is applied. This kind of simulation runs faster than real time to anticipate (multiple) likely future process states Smyth (2007). For this type of problem, a very fast simulator is necessary. In Matics et al. (2005) simulation models are presented covering the temporal and informational ranges of system operation. In simulation-time mode models and simulations can be utilized for the following aims:

Operator training - the goal of the training is to prepare operators for solving standard situations in the controlled system as well as to train actions in the case of unexpected behavior.
Simulating scenarios - based on changing process conditions the several scenarios can be simulated. This approach is especially used when model predictive control is too complex and expensive. The goal of the simulation is not to find the optimal solution but to identify some limits. Results will be used for operator decision.

In the simulation-time operation simulations run as batch process without any synchronization. The proposed framework is oriented on this kind of simulations but the concept and all parts of the framework are designed to allow the run-time extension in the future.

2.2 Data access

Data are needed in each part of the simulation life cycle. There are two basic views of data in simulation tools. In the time scope we distinguish between online and off-line data, both data groups are accessed via different interfaces. Instant values of process variables are online data. When needed, simulators must be connected to SCADA system or OPC servers. Other data are off-line data. These data can be divided in several subgroups:

- historical records of process variables
- process/real system parameters
- model parameters
- simulation parameters

Historical records of process variables are the biggest part of off-line data. In general they can have different properties (like sampling periods, variable type etc.) than online data. The historical records are mainly used in the development phase for the structure and parameter design as well as for the problem analysis. These data could be also used in the operation mode for a simulation initialization.

Process parameters or parameters of real systems are physical features of controlled process like pipe diameters, wire lengths, voltages, current limits etc. These parameters needn't to be directly used in simulators but they can be utilized during the simulation design phase for estimation of model parameters. Different scenarios with prepared sets of process parameters are very often simulated to estimate the system behavior with modified components.

Model parameters are values used in the simulation scheme. A diameter of a suction pump pipe is a process parameter. When the behavior of this pump is represented

by the first order system then the time constant of the transfer function is a model parameter.

Naturally the process itself can have parameters which do not represent physical values but describe the process globally. *Simulation parameters* are usually given by users for particular simulation run, e.g. start and stop time of simulation, solver parameters.

2.3 User

During the simulation life cycle the following userss can join the process:

- Control engineer - control engineer designs a model structure, create data interfaces, and identifies all parameters. He usually works outside of the target platform and this work is mainly focused on the simulation side of the problem.
- Application engineer - in general the man who implements a solution on site can be different from a solution designer/control engineer. This user can perform the same tasks as control engineer but it is usually done in the target environment.
- Operator/Line Manager/Dispatcher - it is a final user running simulation in the restricted modes.

This paper does not deal with the integration of simulations with SCADA systems but our goal is to describe the framework which serves as the design environment and which can be used as the operation environment for a sort of tasks. Such a real project is presented in this paper.

3. STATE OF THE ART

Presently the integration in business IT systems is a standard task which is often run. Enterprise systems are routinely built with integration in mind, e.g. all systems provides interfaces supporting integration process. In the automation systems the situation is very different. Tools within one vendor are sometimes integrated but hardly between vendors. APIs and exchange formats often do not follow established (open) formats.

One of the major issues with the simulation integration is interoperability. Many methods and architectures solving the problem of interoperability and reusing existing simulation tools appears, like HLA, DIS, CAPE-open etc.

The High Level Architecture (HLA) is defined as a general purpose architecture for distributed computer simulation systems. It has been designed to support reuse and interoperability of simulation code. But it suffers from significant drawbacks because it does not support dynamic resource management and sharing. It is a crucial issue because data and system parameters are stored in many forms and storages.

The CAPE-open is a specification defining the interfaces of software components of simulators, see CAPE-Open Specifications. The standard has been developed in the project between 1997 and 2000, currently the last version from 2003 is available. This very detailed standard defines single parts of a process simulator. The concept of CAPE-open respects the structure of industrial processes so the representation of material and energy flows is possible. It means that internal interfaces of simulations are defined to ensure simulation module reuse and exchange. While this standard focuses on model itself, our work deal with the simulation environment and its ability to interact with other tools.

Different usages of process simulations have been widely discussed by researchers in the field of control engineering. But the life cycle of simulations became an important topic in the last decade. The most discussed works (CAPE-Open and HLA) focus on the simulation code reusability and interoperability and their point of view is model-centric. Proposed standards enable replacement of simulation modules and components. Interfaces and communication standards between simulation modules for the run-time mode are defined here. The most complex approach to the simulation life cycle integration is presented in Karhela (2002). The proposed architectural solution covers many user roles and all simulation life cycle but it is not service oriented and modular. Our work should consider the complete life cycle of simulations. Supporting other phases would simplify building simulation-based testing and operation off-line training. Also easy transition between different phases of the simulation development process must be ensured by the environment.

Despite of the fact that SCADA systems are cores of automation control systems, only small effort has been dedicated to the problem of SCADA system integration. In Junior and Pereira (2003) the integration of object-oriented modeling tools with SCADA systems is discussed. An integrated environment with properties that reuse simulation elements in the generation of HMI for supervisory tasks is presented. But this work is mainly focused on the HMI design and simulation code reusability. From the technical point of view the concept and used technologies don't conform to the present requirements. The simulators with SCADA systems are very often used for training purposes. In Hua et al. (2004) the training simulator of the electrical network (located in Henan Electric Power Dispatching and Communication Center) is described. The goal of this approach is not to provide a general integration framework. The compilation of the simulation code into the general executable form is very often used to integrate simulations into the SCADA system Matics et al. (2005). Not all simulators allow compiling the simulation into the executable form and the variability of the usage in the run-time environment is also restricted. More attention has been paid to the integration of SCADA systems and information and management systems. Almost all vendors of SCADA systems provide the utilities and software tools for connecting data systems on the managment level with datastorages in the SCADA system. Thus the research was oriented only on the integration of process specific systems, mainly in power supply engineering Li et al. (2001), Ghoshal (1997), Dragojlovic and Milenkovic (2005), and Phillips et al. (1996).

From the present point of view it is clear that the interoperability cannot be facilitated only on the technical level. Although the semantic level of integration is very important (even more important than the technical level) and our research group works on these topics, this part is not addressed in this paper. The goal is to propose a general service oriented architecture, its components and

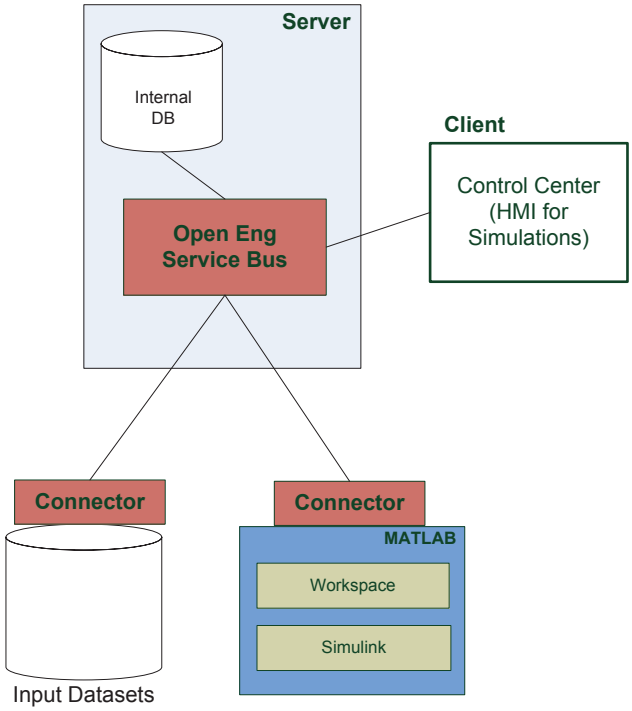

Fig. 1. Framework architecture

features which will provide background for semantics and syntactic integration.

4. SERVICE BUS

A process model is a mathematical description of the system behavior. The process model is usually implemented in a program used for the process modeling and simulations. This software tool or program is called a *simulator*. An experiment with a process model is made to answer questions about the modeled process. *A simulation run* is a particular experiment made with a process model in a simulator.

The technical integration is based on the Open Engineering Service Bus - OpenEngSB. The OpenEngSB is an open source, extensible tool integration platform. The project can be found at www.openengsb.org, Biffl et al. (2009), Biffl and Schatten (2009). The OpenEngSB works on the level of routing messages between the tools. The OpenEngSB consists of tool domains, connectors and core tools, the architecture overview is in Fig. 1.

The standard domains with appropriate connectors for simulators and data storages are necessary for the simulation integration environment.

Data management - domain covering tools for historical data access and management,
User management - user authorization, access restriction, security, user roles etc.,
Version control system - version control system for simulation source code as well as for data and parameter management,
Notification system - it notifies user about workflow statuses, results etc.,

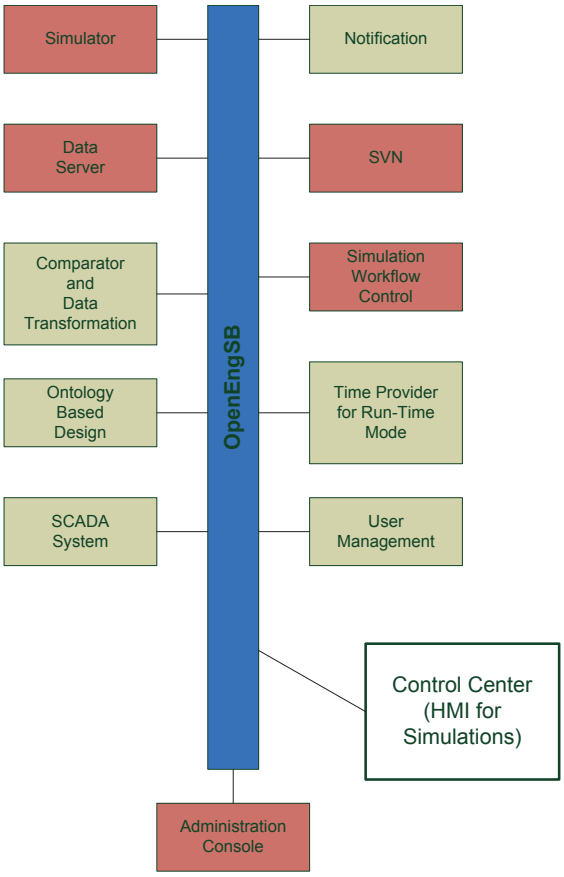

Fig. 2. The simulation framework - domains overview

Parameter management - import and export of parameters, user modification before simulation run, model as well as system parameters.

Beside these domains also domains for source code management, user management, notification and journal system are implemented in order to offer general services. In the following subsections the specific domains providing specific services for simulation integration are described. The domain overview is depicted in Fig. 2.

4.1 Simulation Workflow

Models and simulations usually have more than one task. Therefore the whole solution is spread into several independent modules. This set of simulation modules is called a simulation workspace, all modules regardless of their purpose are part of the workspace. The integration environment provides management of more than one workspace.

A simulation project is a subset of modules designed to solve a particular task. The project specifies topic of the simulation. More projects can be defined in one workspace with different settings or different modules. The project defines workflow of the simulation, it specifies simulation definition within the workspace. The project workflow defines set of modules started/used during the project execution.

A simulation workspace contains all modules including modules for data preprocessing, analysis and initialization. Even if the same simulations modules are used for estimation of unmeasured values in the real-time as well as for for

the dispatcher training, probably both tasks will run in the different synchronization modes and with different input data sets. A special domain is devoted to the simulation workspace management, simulation project definitions and control.

One of the most important aims of the simulation workflow is to detect simulation run with the same attributes. The compared attributes for each simulation run are: simulation source code (its workflow) and its version, selected input data set with its version, values of all types of parameters. The simulation results are not dependent on data values received during the simulation processing from the external data sources or sent into the data sources. The difference can be found in the time of the data requests, but the data shall be the same (either they are requested as actual data or as historical data later). Comparison of the simulation runs are performed for all modules in the simulation workflow. With this approach the computational time can be significantly decreased in the environments with the very complex structure of simulation modules used by more users.

The modules are associated with workspace, because they can be shared between projects (more projects in the workspace can use the same modules). But the overall simulation source code reuse is not ensured in this environment. If required, other approaches must be implemented in the development tools. Each simulation consists of running more modules. The product should provide management and definition of modules for each simulation (e.g. XML file).

The integration platform provides management and definition of simulation workflows and single modules for each simulation. The workflow is defined outside of the simulation tools (one step of the simulation can be simulation of one simulation module). The different versions of workflow are tracked and stored within integrated version control system.

The simulation workflow and its specifications support parallel simulation modules execution. The modules are started at the same time with the same set of parameters (also their values). If such step is performed in the simulation workflow, the next step or section can be started only when all modules performed in parallel are finished.

4.2 Data processing

A data manipulation is a significant portion of total effort. There are several reasons why a special domain is created for data processing. Data processing involves such steps as data filtering, a preliminary analysis during the design phase, a model input selection, a simulation initialization or a result evaluation and testing. Data processing modules can be called at any time independently of simulation modules, e.g. only for graphical presentation of results. Also the periodicity of calling is very different. For instance input selection methods or preliminary analysis are done one times during the design process but the result evaluation and testing is proceeded always. The last reason for creating the special domain is the fact that some simulation tools does not contain any instruments for data processing.

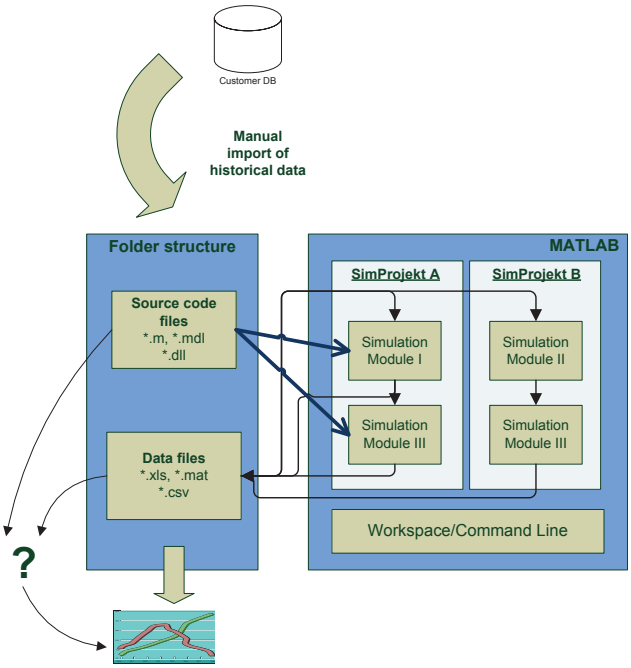

Fig. 3. Real example - Original structure

4.3 Ontology-based design

As already mentioned this paper is addressed to the technical part of the integration. From the beginning it is obvious that the integration process cannot be solved only at the technical (software) level. At this level it is mainly possible to ensure the data exchange. But to design the semantic level of integration another approach must be chosen. We selected the ontology driven approach for the framework design. Our experience show that all tools taking part in the ontology-based design should be available from the dedicated domain.

5. REAL PROJECT

Secure operation of a power transmission system is the main goal targeted by a transmission system operator. The task of the optimal planning of the transmission system operation is solved for Czech TSO (ČEPS), the complex approach to this task is presented in Havel et al. (2008). In the last few years several simulation models have been developed to help solving the problem Černý et al. (2008). It consists of modules implemented in Matlab/Simulink programming environment, external solver for optimization tasks etc. The main research was focused on algorithms, modeling and simulations and it necessarily brought changes in used solvers and simulation engines. For example an implementation of a particular part of model has been changed from an analytical model to a rule based fuzzy model. Such a change is time consuming because new interfaces and new object models must be created.

This project represents the class of problems where no synchronization is required. Presently the data sets are available at the local data storage but it is expected that in the future data will be automatically downloaded from the process storages. Also the single user approach has changed to multi-user approach. With the growing

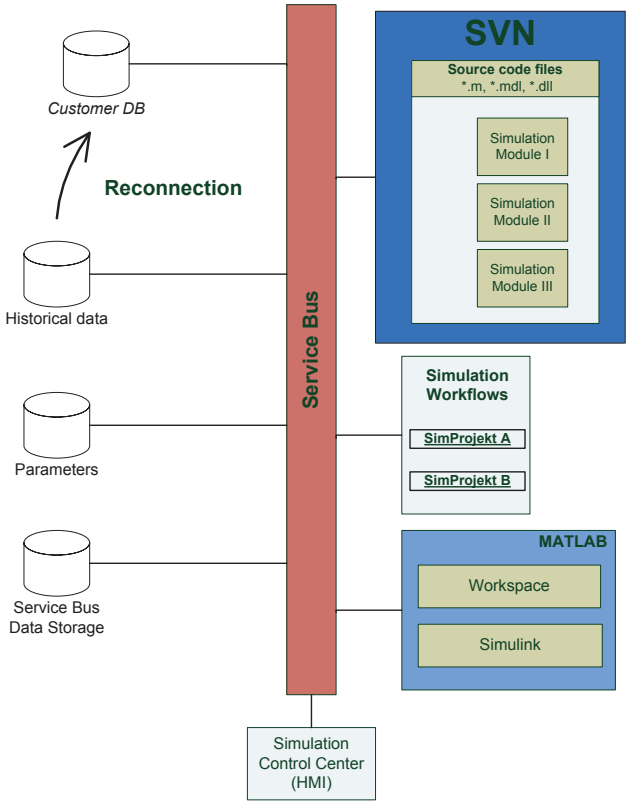

Fig. 4. Real example - Structure with Service Bus

capabilities more projects in the simulation workspace was created by different users. One of the most critical problems is keeping information on single simulation runs. Without the simulation framewok is hard to relate results with input data, parameters and simulation source code. It must work without user intervention.

6. CONCLUSIONS AND FUTURE WORK

In this paper we presented the concept of the simulation integration framework. The goal is to provide an environment that connects simulations not only in the run-time mode but also support the complex life cycle of simulations. Because of complexity the work has been divided into two independent phases. In the first phase only simulation time simulations are supported. No synchronization and no connection to SCADA system is requested. The presented framework integrates components in office-like design environment.

From the experience with the initial prototype we found that the following issues must be addressed in the future work:

- workflow design
- support of parallel simulations
- finer fragmentation of testing scenarios
- transition between design and testing phases

In the near future the second phase will be started. The main goal of the second phase is to integrate simulations with SCADA systems and other parts of the control system. It will require synchronization between real system and tools and thus new services and domains for the run-time mode are necessary. For the run-time mode completely new requirements regarding operation speed, deterministic behavior and performance are demanded.

ACKNOWLEDGEMENTS

This work has been supported by the Christian Doppler Forschungsgesellschaft and the BMWFJ, Austria.

REFERENCES

Biffl, S. and Schatten, A. (2009). A platform for service-oriented integration of software engineering environments. *Proc. of the 8th International Conference on Software Methodologies, Tools and Techniques (SOMET 09)*, 75 – 92.

Biffl, S., Schatten, A., and Zoitl, A. (2009). Integration of heterogeneous engineering environments for the automation systems life cycle. *Proc. IEEE Industrial Informatics (IndIn) Conf., IEEE Computer Society Press, 2009.*

CAPE-Open Specifications (2003). URL `www.global-cape-open.org`.

Dragojlovic, S. and Milenkovic, O. (2005). Integration of scada system in technical information system. *18th International Conference and Exhibition on Electricity Distribution*, 1–3.

Ghoshal, K. (1997). Distribution automation: Scada integration is key. *IEEE Computer Applications in Power*, 10, 31–35.

Havel, P., Horáček, P., and Fantík, J. (2008). Optimal planning of ancillary services for reliable power balance control. *IEEE Transactions On Power Systems*, 23(3), 1375–1382.

Hua, B., Zhou, J., and Yu, J. (2004). Integration of exist scada/ems with dispatcher training simulator system. *IEEE PES Power Systems Conference and Exposition*, 2, 829– 838.

Junior, W. and Pereira, C. (2003). A supervisory tool for real-time industrial automation systems. *Sixth IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, 230– 237.

Karhela, T. (2002). *A Software Architecture for Configuration and Usage of Process Simualtion Models.* Ph.D. thesis, Helsinky Universisty of Technology.

Li, X., Gao, M., Liu, J., Ding, Z., and Duan, X. (2001). A software architecture for integrative utility management system. *IEEE Power Engineering Society Winter Meeting*, 2, 476–480.

Matics, J., Krost, G., Roggatz, C., and Spanel, U. (2005). Dispersed generation modeling in scada time scale. *IEEE Proceedings of PowerTech*, 1 – 7, 27–30.

Phillips, N., Gann, J., and Irving, M. (1996). The simian architecture - an object-orientated framework for integrated power system modelling, analysis and control. *Fourth International Conference on Power System Control and Management*, 148–153.

Smyth, E. (2007). Scada and telemetry in gas transmission systems. *ABB White Paper.*

Černý, V., Fantík, J., and Janeček, E. (2008). Monte-carlo simulation of electricity transmission system operation. *in Proceedings 17th IFAC World Congress.*