TOWARDS A COMPONENT-ORIENTED DESIGN OF MODELING AND SIMULATION TOOLS

Michael Syrjakow, Elisabeth Syrjakow

Institute for Computer Design and
Fault Tolerance (Prof. D. Schmid)
University of Karlsruhe
76128 Karlsruhe, Germany.

Helena Szczerbicka

Institute for Computer Science
Fachbereich Mathematik und Informatik
University of Hanover
30167 Hanover, Germany.

ABSTRACT

Over the last years component technology has deeply penetrated the area of computer simulation. As a result a new research field has been initiated, which roughly can be divided up into the following two sub-areas: component-oriented development of simulation models and component-oriented development of modeling and simulation (M&S) tools. Most research activities today focus on the development of component-oriented simulation models. For that purpose a global standard called High Level Architecture (HLA) was adopted by the US Department of Defense. Component-oriented development of M&S tools however has been neglected for the most part. This is a very unsatisfying situation because many M&S tools still have a monolithical software design which is difficult to maintain and to extend and which doesn't correspond any more with the modern distributed Web-centered technologies of today. The main focus of this paper will be on a hybrid integration approach for M&S tool components being a combination of loose document-based and tight invocation-based integration concepts. Concerning this, we report about the current research work on a pragmatic XML-based model interchange format for High-Level Petri Nets and we elaborate on the design of powerful experimentation components and their integration into M&S tools.

1 INTRODUCTION

Complexity of computer software is constantly growing, both in the size of developed systems, and in the intricacy of its operations. This general observation particularly applies to simulation applications, which have grown enormously over the past decades. Today the most prominent approaches to master the complexities of large-scale software development are object-orientation and component technology, which usually is built up on object-orientation. Component approaches concentrate design efforts on defining interfaces to pieces of a system and describing an application as the collaborations that occur among those interfaces. Implementers of a component can design and build the component in any appropriate technology as long as it supports the operations of the interface and is compatible with the component execution environment. For that reason the interface is focal point for all analysis and design activities of component-based software development (Brown 2000, Szyperski 1999). Component technology has also deeply influenced the area of computer simulation. Here we can distinguish two main fields of activity: component-oriented development of simulation models and component-oriented development of modeling and simulation (M&S) tools.

For a component-oriented development of distributed simulation models the US Department of Defense (DoD) Modeling and Simulation Office (DMSO) has adopted a global standard called High Level Architecture (HLA). Its chief intent is to allow simulation applications to be created by combining other simulations. Such a combined simulation is called a federation. The constituent simulations are called federates. The basic idea of the HLA is the strict separation of the simulation functionality from the infrastructure for interoperability. Altogether the simulation interoperability standard HLA is defined by: 1.) rules which govern the behavior of the overall distributed simulation (federation) and their members (federates), 2.) an interface specification, which prescribes the interface between each federate and the Runtime Infrastructure (RTI) providing communication and coordination services to the federates, and 3.) an Object Model Template, which defines how federations and federates have to be documented. Even though the HLA has a military origin the architecture is also very suitable for civilian applications. A detailed description of the HLA can be found in (Kuhl et al. 2000).

In contrary to the area of component-oriented development of simulation models where a standard is available today and where a variety of research activities can be observed, the field of component-oriented development of M&S tools yet remains rather untouched. This is a very unsatisfying situation because many M&S tools still have a monolithical software design which is difficult to maintain and to extend and which doesn't correspond any more with the modern distributed Web-centered technologies of today. In order to illustrate this unsatisfying situation in greater detail we take a look at some existing and widely used M&S tools. We focus on Petri Net tools because they are a quite suitable example to explain the disadvantages of a monolithical software design. It should be mentioned for fairness that these observations also apply to other prominent classes of M&S tools for example Queuing Network (Chen and Yao 2001) tools.

Having surveyed the software design of existing Petri Net tools in Section 2 a hybrid integration approach for M&S tool components is presented in Section 3. Subsequently in Section 4 the requirements for an XML-based model interchange format for High-Level Petri Nets are discussed. Section 5 focuses on the design and implementation of experimentation components. Finally, in Section 6 we summarize and draw some conclusions.

2 DISADVANTAGES OF M&S TOOLS WITH A MONOLITHICAL SOFTWARE DESIGN

Today about 100 different Petri Net tools are available. A comprehensive and up-to-date database can be found at http://www.daimi.au.dk/PetriNets/tools/. Altogether these tools offer 76 different graphical Petri Net editors, 50 different token game animations, 52 different implementations for structural analysis, and 39 different implementations for performance analysis. This variety on principal is not bad. The monolithical software design however makes it almost impossible to integrate for example an outstanding Petri Net evaluation module into a tool with a nice graphical Petri Net editor. Beyond that, all these tools are difficult to maintain and to extend. Another great disadvantage is the lack of interoperability. A user who has edited a Petri Net with one tool usually cannot analyze this Petri Net with another tool. The reasons for that incompatibility are the following: Every Petri Net tool uses its own proprietary file format and often supports only a special kind of Petri Net version. To overcome this unsatisfying situation international standards have to be established concerning

- a mathematical semantic model, an abstract mathematical syntax, and a graphical notation for High-Level Petri Nets. The standards group of the International Organization for Standardization (ISO) relevant for the Petri Nets standardization effort is called ISO/IEC JTC1/SC7/WG11. An overview of the current activities of that group is available at http://www.daimi.au.dk/PetriNets/standardisation/#sc7 resources.
- a general Petri Net interchange format that supports all features of existing and forthcoming Petri Net tools. An overview of the ongoing standardization efforts of an XML-based Petri Net interchange format is given in Section 4.
- a component architecture for M&S tools. In addition to the two standards mentioned above an appropriate component architecture for M&S tools is of great importance. In this paper the main focus will be on such a component architecture which is described in detail in the following Section.

3 A HYBRID INTEGRATION APPROACH FOR M&S TOOL COMPONENTS

In contrary to the HLA which provides with the RTI a very demanding infrastructure for a tight coupling of highly interdependent simulation components a component architecture for M&S tools should also support a much looser component coupling. This is justified because M&S tools usually consist of a very limited number of quite self-sufficient and coarse-grained components. From the user's point of view usually the following software parts can be identified within an M&S tool

- model editor
  A model editor allows the modeller to edit new and to modify existing models. Model editors may exist in several variations. We can distinguish textual and graphical editors. Modern Web-based modeling tools may allow collaborative online editing of models. A model editor basically can be realized as an independent stand-alone component. Its output is a model description in a specific description format which is characterized by the supported modeling technique.
- model analysis/evaluation modules
  These modules are used to analyse and to evaluate models generated by the model editor. In case of High-Level Petri Nets (Jensen 1991) we can distinguish between a mathematical analysis of structural properties (place-invariants, transition-invariants, boundedness, etc.) and performance evaluations (stationary analysis, transient analysis). Performance evaluation can be computed either analytically or by simulation. An evaluation module may provide some animation features. In case of Petri Nets a token game animation for example.
- experimentation modules
  These modules are optional. They allow goal-driven experimentation with a model, for example to find optimal parameter settings, determine sensitive model parameters, perform a model validation, etc. To fulfill these tasks usually a lot of model evaluations (experiments) are required. For that reason experimentation modules should be closely coupled with evaluation modules.

Fig. 1 shows the different M&S tool components and their interdependencies. As we have described above the collaboration of these components is based on two kinds of interactions: exchange of documents and invocation of model evaluation modules. For that reason an obvious and pragmatic integration approach for M&S tool components is a hybrid one being a combination of loose document-based and tight invocation-based integration techniques. For remote invocations universal component 'wiring' standards like CORBA (Common Object Request Broker Architecture), EJB (Enterprise JavaBeans) or DCOM (Distributed Component Object Model) can be used. A specialized standard like the HLA, which focuses on the special needs of tightly coupled simulation models (federation management, time management, etc.) is not needed in this case. For document-based integration standardized document interchange formats are required. Here XML-based approaches are the most promising ones.

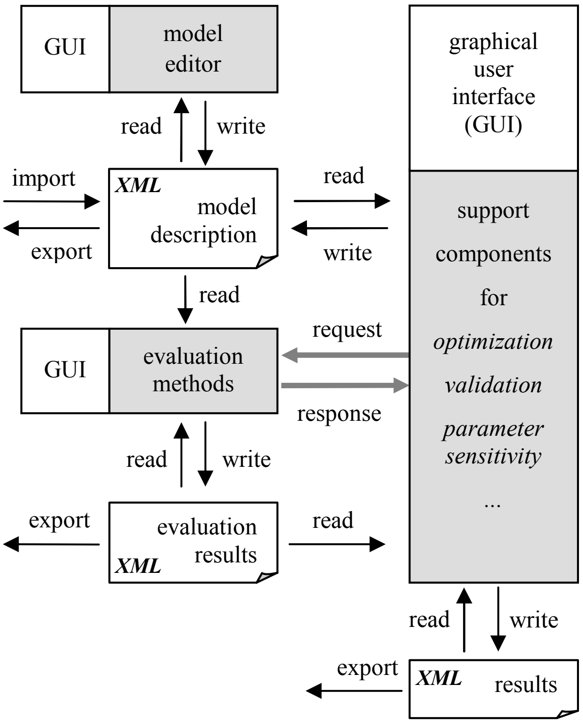

Figure 1: M&S tool components and their interdependencies

The advantages of the hybrid integration approach described above are manifold: 1.) It enables a flexible distribution of the involved components within a computer network. 2.) It allows user access by traditional application clients or by Java-based Web clients. 3.) It enables an easy integration of existing monolithical tools as a whole by transformation of the proprietary model description format into a standardized XML-format or partially by appropriate component wrappers. 4.) It considerably eases tool modifications and extensions. 5.) It represents a good basis for agent-based approaches. 6.) Beside all these technical advantages component-orientation opens several economic and organizational advantages (software reuse, clear separation of responsibilities, etc.).

Fig. 2 shows a possible realization of a component-oriented M&S tool based on a distributed 4-tier architecture. The first tier contains client components which allow access (Web- or application-based) to server components residing on the other tiers behind. The application server contains the M&S tool components shown in Fig. 1. For their component-oriented realization several component models can be applied for example EJB, DCOM, CORBA, etc. Persistent modeling data are saved on a database-server representing the fourth tier of the distributed architecture.

Finally, it should be mentioned that our component approach is not only restricted to Petri Net tools but can be used for all kinds of M&S tools. In the following two Sections we will explain more detailed two important sub areas of our approach: 1.) An XML-based interchange format for models of a specific modeling technique (in our case Stochastic Petri Nets) and 2.) experimentation components allowing the modeller to automatically extract information about the behaviour of complex simulation models. The presented methods and concepts have been already successfully used for the prototypical realization of a component-oriented Stochastic Petri Net (SPN) M&S tool.

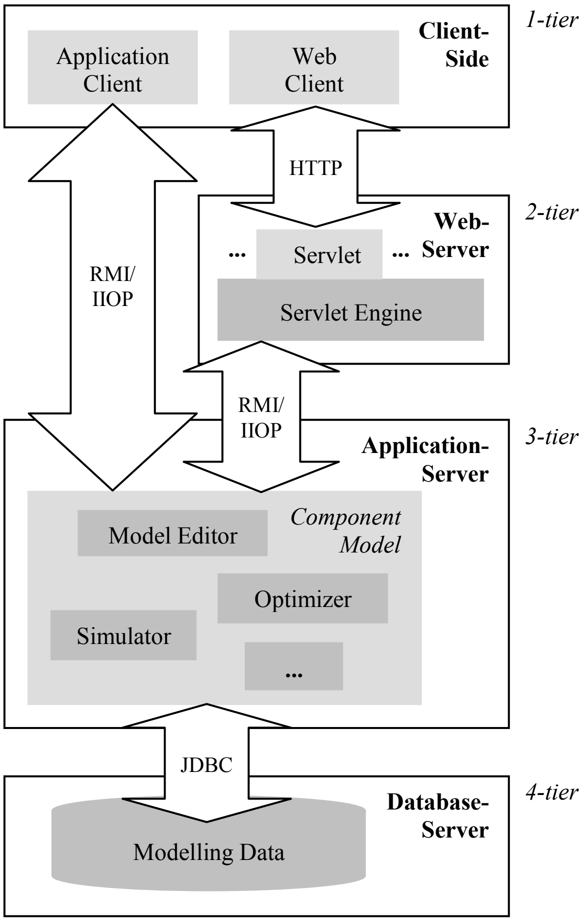

Figure 2: Possible realization of a component-oriented M&S tool based on a distributed 4-tier architecture

4 AN XML-BASED MODEL INTERCHANGE FORMAT FOR HIGH-LEVEL PETRI NETS

The idea of a standardized interchange format for Petri Nets is not new. However, the attempts to define such a standard file format were not very successful in the past. The main reasons for that failure are the following:

1.) Each Petri Net tool usually supports its own special version of Petri Nets.

2.) As a consequence of this each Petri Net tool provides a special file format which solely satisfies the needs of the supported Petri Net version.

3.) The lack of appropriate description techniques being flexible enough to cover both the common essence of all existing Petri Net types and beyond that, the special features of any particular Petri Net extension.

Recently however, the idea of a standardized Petri Net interchange format got a new boost due to the availability of the Extended Markup Language (XML). Today XML seems to have the power to become a mayor means for a homogeneous and standardized exchange of information. XML allows the specification of specialized markup languages for the convenient exchange of information in specific areas of research or business. Examples of recent markup languages based on XML are the Chemical Markup Language (CML), the Mathematical Markup Language (MathML) or the Astronomical Instrument Markup Language (AIML).

In the area of Petri Nets several research groups are currently working on an XML-based model interchange format which of course should be based on the ISO/IEC Petri Net standard. Beyond that, this format should be generic and extensible to be able to cover all existing and forthcoming Petri Net classes. A preliminary proposal for such an interchange format can be found in (Jüngel et al. 2000). The proposed format consists of two parts:

1.) A general part called Petri Net Markup Language (PNML) which captures the common features of all existing Petri Net versions.
2.) A specific part called Petri Net Type Definition (PNTD) which allows to specify additional features. This part is of great importance because it provides openness for future Petri Net extensions.

In our research work we are currently working on a PNTD for Stochastic Petri Nets (SPN) (Lindemann 1998). An overview of the ongoing standardization efforts of an XML-based Petri Net interchange format can be found at http://www.oasis-open.org/cover/xmlAndPetriNets.html.

As shown in Fig. 1 appropriate description formats for models and modeling results are an integral part of our proposed M&S tool architecture. They allow a simple document based integration of tool components which is usually much easier to realize than invocation-based approaches.

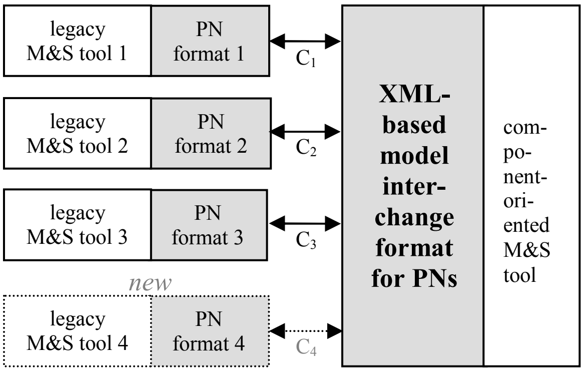

Figure 3: Necessary format conversions with an XML-based model interchange format

Beyond that, existing monolithically designed M&S tools can be easily incorporated into our architecture without any expensive software modifications. For that purpose only appropriate file converters (C) have to be realized being able to convert the proprietary file formats of the legacy tools into the XML-based model interchange format (see Fig. 3). This has been proven to be a very simple way to achieve compatibility between several legacy Petri Net tools allowing the mutual use of parts (editors, evaluation modules, etc.) of them. As shown in Fig. 3 for the incorporation of a new legacy tool the realization of only one additional file converter is required. Without such a standardized interchange format the number of required file converters would not increase linearly but quadratically. As indicated in Fig. 4 for n different file formats $(n^2\text{-}n)/2$ file converters would be required to achieve compatibility between the n corresponding PN tools.

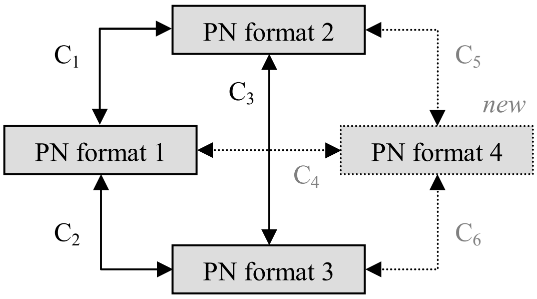

Figure 4: Necessary format conversions without an XML-based model interchange format

5 EXPERIMENTATION COMPONENTS

M&S tool developers often neglected or in the worst case just omitted experimentation components in the past. This was mainly caused by the monolithical software design of the existing M&S tools which made a later integration of additional experimentation functionality rather intricate and expensive. With the enormous increase of model complexity however these components have gained great importance because experimentation goals like finding optimal or sensitive model parameters cannot be reached by hand any more. Following our hybrid integration approach it is very easy to supply an M&S tool with additional functionality for experimentation. In the following we take a detailed look at a parameter optimization component, which provides efficient and universally applicable methods to optimise the behaviour of complex simulation models.

For that demanding task classical analytical optimization methods often cannot be applied any more because it is usually not possible to find a mathematical representation of the underlying goal function. In such cases only direct search methods are applicable which do not use any additional analytical information like gradients etc. about the goal function. To guide the optimization process only goal function values are required. Because of that property direct search methods are universally applicable to any kind of optimization problem.

Direct optimization strategies work iteratively. Outgoing from a usually randomly generated starting solution the goal is to find a solution which maximizes or minimizes a goal function being either one or a composition of several model outputs. Because the evaluation of a simulation-based goal function usually requires considerable computational resources the optimization goal should be reached with a minimum number of iteration steps. Fig. 5 shows the iterative process of direct model optimization. The specific problems and requirements of model optimization are comprehensively described in (Syrjakow and Szczerbicka 1997).

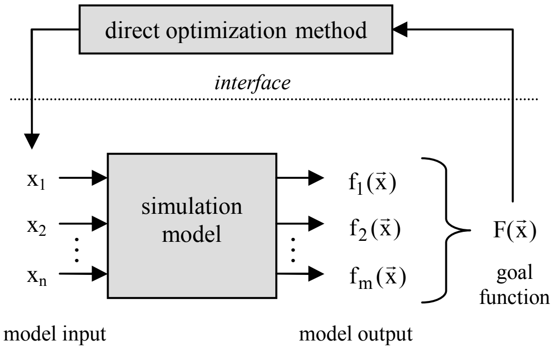

Figure 5: Direct optimization of a simulation-based goal function

To automate the process of model optimization a specific interface is required. Its task is to couple the optimization process with the process of model simulation allowing data exchange as well as process synchronisation. Fig. 6 shows the interactions of a direct optimization component with a model evaluation component in more detail. For specification of the optimization problem the optimization component has access to two files: the model description and the evaluation results. The model description comprises all existing model parameters allowing the user to select the parameters which have to be optimized. To define the goal function the user has to select one or to combine several model outputs which can be found in the evaluation results file.

In each iteration step of the optimization process the direct search strategy generates a vector of parameter values which are entered into the model description. Subsequently the optimization component sends a request to the evaluation component containing several evaluation parameters. In case of a simulation component for example the simulation run length, kind of confidence interval method etc. has to be defined. After model evaluation the evaluation component sends a response message to the optimization component to indicate that the required model outputs have been calculated and are now available in the evaluation results file. Outgoing from these outputs the optimization strategy generates a new parameter vector. This alternating process continues until a termination criterion is fulfilled.

Summing up, the extension of an M&S tool by an optimization component requires in addition to the exchange of standardized documents also a more tight invocation-based integration concept. This is unavoidable because the two alternating processes of optimization and model evaluation have to be synchronized to each other.

For realization of the required component invocations universal component 'wiring' standards like CORBA, EJB or DCOM can be used. For our component-based Petri Net tool prototype we have used CORBA because it provides the following main advantages

- Programming-language independent interface
  Interfaces between clients and servers are defined in a standardized Interface Definition Language (IDL).
- Easy legacy integration
  Using IDL, programmers can encapsulate existing applications in wrappers and use them as objects on the ORB (Object Request Broker).
- Rich distributed object infrastructure
  Distributed applications require more functionalities than simple method invocations. CORBA offers them a rich set of distributed object services and facilities.

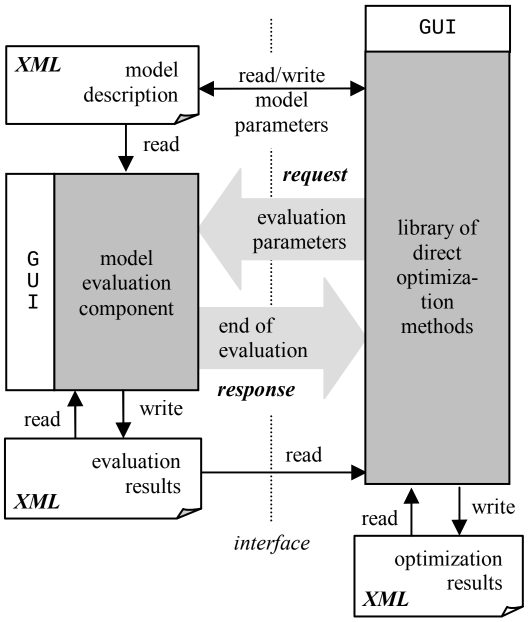

Figure 6: Interactions between the components for model evaluation and optimization

Finally, we will take a more detailed look at the direct optimization strategies which are offered by our optimization component. Table 1 gives an overview of these strategies. Beside common direct strategies for global and local search our optimization component offers hybrid and so called multiple stage optimization methods. Hybrid strategies are combinations of global and local search methods. Here the goal is to exploit their advantages, i.e. to get both high optimization success and high quality results. Beyond that, our hybrid optimization strategies have been proven to be very efficient. This is extremely important for model optimization because a goal function evaluation may be very time consuming. For that reason the optimization strategy should require only a very limited number of goal function evaluations to achieve a good result. A multiple stage optimization strategy allows to execute atomic and hybrid search methods several times in order to localize not only one but a sequence of important extreme points of a goal function.

kind	name
global	Genetic Algorithms (GA), Simulated Annealing (SA)
local	Hill Climbing Strategies (HC)
hybrid	GA+HC, SA+HC
multiple stage	multiple execution of one of the optimization strategies listed above allowing a systematic search for the most prominent extreme points of a given optimization problem

Table 1: Overview of the optimization strategies offered by our optimization component

To prevent that previously found optimum points are localized again in subsequent iteration steps a method called avoidance of reexploration is applied. Altogether our optimization component offers a powerful modular assembly system of direct optimization strategies which is realized component-oriented itself. A detailed description of our optimization algorithms can be found in (Syrjakow and Szczerbicka 1997, Syrjakow and Szczerbicka 1999).

6 CONCLUSIONS

In this paper we presented a hybrid integration approach for M&S tool components being a combination of loose document-based and tight invocation-based integration concepts. Core of our approach is an XML-based model interchange format, which allows a homogeneous and standardized information exchange between tool components. For the tight coupling of tool components universal component 'wiring' standards are used. Our integration concept has been proven to be very flexible and applicable to all kinds of M&S tools. For its validation we have applied it to realize a component-oriented SPN modeling tool. A great advantage of M&S tools with a component-orientated software design is their openness for all kinds of extensions. As a result tool developers can fully concentrate on the development of such extensions and are not any longer needlessly stressed with their integration. Today especially experimentation components are of great importance because they allow to automatically extract valuable information about the behaviour of complex simulation models which isn't possible by hand any more.

ACKNOWLEDGEMENTS

We want to thank Prof. D. Schmid for his encouragement and support of our work. We also thank our students, especially Sabine Schillinger and Florian Schmidt for their engagement and contributions.

REFERENCES

Brown, Alan W. 2000. Large-Scale, Component-Based Development. Prentice Hall.

Chen, Hong and David D. Yao. 2001. Fundamentals of Queuing Networks - Performance, Asymptotics, and Optimization. Springer.

Jensen, Kurt. 1991. High-Level Petri Nets: Theory and Application. Springer.

Jüngel, M., E. Kindler, and M. Weber. 2000. The Petri Net Markup Language. In Proceedings of the AWPN-Workshop, Koblenz, Germany.

Kuhl, F., R. Weatherly, and J. Dahmann. 2000. Creating Computer Simulation Systems - An Introduction to the High Level Architecture. Prentice Hall.

Lindemann, Christoph. 1998. Performance Modelling with Deterministic and Stochastic Petri Nets. Wiley.

Syrjakow, M. and H. Szczerbicka. 1997. Efficient Methods for Parameter Optimization of Simulation Models. In Proceedings of the 1st World Congress on Systems Simulation, Singapore, Republic of Singapore, pp. 54-59.

Syrjakow, M. and H. Szczerbicka. 1999. Efficient Parameter Optimization based on Combination of Direct Global and Local Search Methods. In Evolutionary Algorithms (Ima Volumes in Mathematics and Its Applications, Vol. 111), L.D. Davis, K. De Jong, M.D. Vose, L.D. Whitley (eds.), Springer Verlag New York, pp. 227-249.

Szyperski, Clemens. 1999. Component Software - Beyond Object-Oriented Programming. Addison-Wesley.

AUTHOR BIOGRAPHIES

Michael Syrjakow was born in 1964 in the Federal Republic of Germany. He received the Dipl.-Inform. degree from the University of Karlsruhe, Germany in 1991. Since then he has been with the professional group Modeling and Simulation at the Institute for Computer Design and Fault Tolerance at the University of Karlsruhe. In February 1997 he received the Ph.D. in Computer Science from the University of Karlsruhe. His Email and Web addresses are <syrjakow@ira.uka.de> and <goethe.ira.uka.de/~syrjakow/>.

Elisabeth Syrjakow was born in 1970 in the Federal Republic of Germany. She received the Dipl.-Inform. degree from the University of Karlsruhe, Germany in 1999. Since then she has been with the professional group Modeling and Simulation at the Institute for Computer Design and Fault Tolerance at the University of Karlsruhe. Her Email and Web addresses are <lisa@ira.uka.de> and <goethe.ira.uka.de/~lisa/>.

Helena Szczerbicka was born in Poland. She received the M.Sc. in applied Mathematics and the Ph.D. in Computer Science from the Technical University of Warsaw, Poland, in 1974 and 1982, respectively. In July 1985 she joined the Faculty of Computer Science at the University of Karlsruhe, Germany. In May 1994 she became a professor in computer science at the University of Bremen, Germany. Since May 2000 she has been a professor at the University of Hanover, Germany. Her Email and Web addresses are <hsz@informatik.uni-hannover.de> and <www-a2.informatik.uni-hannover.de/~xmb/helena.html>.