TimeNET 3.0 Tool Description

Armin Zimmermann Reinhard German Jörn Freiheit Günter Hommel

Prozeßdatenverarbeitung und Robotik
Technische Universität Berlin, Sekr. FR 2-2
Franklinstr. 28/29, 10587 Berlin, Germany
{ azi | rge | freiheit | hommel }@cs.tu-berlin.de

Abstract

This paper presents a new version of TimeNET, a software tool for the modelling and performability evaluation using stochastic Petri nets. The tool has been designed especially for models with non-exponentially distributed firing delays. A general overview of the software package is given in the paper as well as a description of new features.

1 Introduction

Model based performance evaluation of technical systems is a powerful and inexpensive way of predicting the performance before the actual implementation. Petri nets are a graphical method for the convenient specification of discrete event systems. They are especially useful for systems with concurrent, synchronised, and conflicting activities. Since the modelling framework of stochastic Petri nets has been proposed, many algorithms and their implementations as software tools have been developed.

This paper describes a new version of the tool TimeNET, which is especially tailored for the performability evaluation of stochastic Petri net models with non-exponentially distributed firing times. Transient and steady-state analysis and simulation components are provided. A new graphical user interface and an environment for the modelling and evaluation of manufacturing systems with coloured Petri nets have been added recently. Another addition is the possibility of analysing models with a discrete time scale. TimeNET has been successfully applied during several modelling and performance evaluation projects.

The development of TimeNET has been influenced by other software tools like GreatSPN [4], SPNP [5], and UltraSAN [21]. The first version of TimeNET [12] was a major revision of the tool DSPNexpress [19], which had been developed at TU Berlin since 1991.

2 Supported Net Classes and Analysis Methods

The software package TimeNET offers two different modelling and evaluation environments. Extended deterministic and stochastic Petri nets with uncoloured tokens and their corresponding evaluation techniques are described in subsection 2.1. The part of the tool dealing with coloured Petri nets is presented in subsection 2.2. Coloured models can contain the same stochastic extensions like eDSPNs.

2.1 Extended Deterministic and Stochastic Petri Nets

Many classes of stochastic Petri nets (SPNs) with different modelling power were developed since their first proposal. Firing delays of transitions are often exponentially distributed due to their analytical simplicity. However, the assumption of a memoryless firing delay distribution is not realistic in many cases and can lead to significant differences for the computed measures. Transitions with deterministic or more generally distributed firing delays are needed for the modelling of systems with clocking or fixed operation times. TimeNET supports extended deterministic and stochastic Petri nets (eDSPNs, [6]), also referred to as Markov regenerative SPNs. The firing delay of transitions in eDSPNs can either be zero (immediate), exponentially distributed, deterministic, or belong to a class of general distributions called *expolynomial*. Such a distribution function can be piecewise defined by exponential polynomials and has finite support. It can even contain jumps, making it possible

to mix discrete and continuous components. Many known distributions (uniform, triangular, truncated exponential, finite discrete) belong to this class.

In the first step of each analysis algorithm structural properties are computed. They can be displayed and checked by the modeller to examine the correct model specification.

2.1.1 Steady-state analysis

For a steady-state or transient analysis of a stochastic Petri net model of any kind, the reachability graph is computed next. Some structural properties are exploited for an efficient generation of the reachability graph [2, 3]. The problem of complex reachability graphs for models of realistic size is known as *state explosion*. TimeNET offers two variants of the reachability graph generation algorithm. One is optimised for speed, while the other uses less memory space. Therefore, bigger models can be evaluated with the second algorithm if the memory available on a workstation is insufficient for the first one. Subnets of immediate transitions are evaluated in isolation, following the ideas presented in [1].

The steady-state analysis component of TimeNET is applicable for GSPNs, DSPNs, and eDSPNs [9]. The reduced reachability graph of a GSPN is isomorphic to a continuous-time Markov chain, because of memoryless state changes. Only the corresponding linear system of equations has to be solved. In case of an eDSPN (and DSPNs as a special case), an additional step is required. The underlying stochastic process is only memoryless at some instants of time, called *regeneration points*. If a transition with non-exponentially distributed firing delay is enabled in a marking, the next regeneration point is chosen after firing or disabling this transition. The time of firing the next exponential transition is taken otherwise.

By taking only the regeneration points into account, a discrete-time Markov chain is embedded. For the later analysis of this Markov chain, the stochastic matrix **P** of one-step transition probabilities and a matrix **C** of conversion factors have to be computed [6]. **P** describes the probabilities of state changes of the embedded Markov chain between two regeneration points. **C** describes the conditional sojourn times in the states between two regeneration points. For states with solely exponential transitions enabled, only the diagonal entry of the corresponding **C** matrix row is different from zero. It contains the mean sojourn time in that marking, which can be computed directly from the reduced reachability graph.

In order to compute the entries of the **P** and **C** matrix, the evolution of the stochastic process during the enabling of a transition with non-exponentially distributed firing delay is analysed. At most one transition of this type can be enabled per marking for this type of analysis. Therefore only exponential transitions may fire during the enabling period, resulting in a continuous-time subordinated Markov chain (SMC) of the non-exponential transition. The transient and cumulative transient solution of this Markov chain computes the **P** and **C** matrix entries. TimeNET uses Jensens method (also known as randomisation or uniformisation) for both solution parts.

In case of an expolynomial firing delay distribution, the algorithm for the transient SMC analysis has to be extended. It differs from the DSPN algorithm only in the computation of the **P** and **C** matrix. The **P** matrix entries require the computation of the integral of the matrix exponential of the generator matrix with respect to the probability distribution function of the non-exponential firing delay, where in the DSPN case a simple integral was sufficient. The average sojourn times in the states of the SMC until regeneration (the **C** matrix entries) are computed as the matrix exponential multiplied with the complement of the probability distribution function, integrated from zero to infinity. Jensens method has been generalised for the efficient numerical computation of the integrals and implemented in TimeNET for expolynomial firing delays [6, 9].

A linear system of equations based on the **P** matrix has to be solved for all steady-state analysis techniques. TimeNET uses standard algorithms like successive over relaxation (SOR) and sparse Gaussian elimination. The state probabilities of the actual stochastic process can then be obtained as the mean sojourn time in each state between two regeneration points. Formally, this corresponds to multiplying the EMC solution vector by **C** and normalising it.

Finally, the token probability distribution in the places of the net as well as the user-defined performance measures are calculated from the state probabilities.

2.1.2 Transient Analysis

The transient analysis of a GSPN model requires the computation of the reduced reachability graph. Jensens method is applied for the transient solution in a second step. TimeNET also provides a compo-

nent for the transient analysis of DSPNs. The method is based on supplementary variables, which capture the elapsed enabling time of transitions with non-exponentially distributed firing delays. State equations can be derived for the description of the dynamic behaviour of DSPNs, consisting of partial and ordinary differential equations which are combined with initial and boundary conditions. This method was presented in [13], and the first implementation of the analysis component is described in [11]. The algorithm has been redesigned in order to improve its efficiency, both in terms of time and memory space. Furthermore, the numerical quality is enhanced and a previous restriction is removed. Automatic step size control is facilitated together with an enhanced error control. The new component is described in [16]. During the transient analysis, TimeNET shows the evolution of the performance measures from the initial marking up to the transient time graphically.

2.1.3 Simulation

TimeNET comprises an efficient simulation component, which evaluates eDSPN models (in steady state or transient case) without the restriction of enabled non-exponential transitions. Before a simulation run is started in TimeNET, the user specifies the desired accuracy of the performance measures in terms of the confidence interval and the maximum relative error of the final result. The initial transient phase of the simulation run of a steady-state evaluation is detected and ignored. Variance estimation of the samples is performed by spectral variance analysis, allowing a robust estimation even for correlated samples as they are common if only one replication of the simulation process is running.

The length of a simulation run is decreased with a parallelisation of simulation processes. Each one of them simulates the whole net and sends sample packets to a central process. As long as the model can be handled on one workstation, this approach is simpler to implement and more efficient than parallel simulation with a distributed model. The central process monitors the accuracy and stops the simulation after reaching the specified threshold. To reduce the simulation length further, variance reduction with control variates is applied [17]. The correlation between an estimator of interest and another stochastic parameter of the model is exploited to reduce the variance of the estimator for the same number of samples. A well-known problem of simulation is the number of samples necessary if rare events with a significant influence on the performance measures occur in the model. TimeNET uses the RESTART method [18] (repetitive simulation trials after reaching thresholds), which belongs to the class of importance sampling techniques.

2.1.4 Models with Discrete Time Scale

A recent addition to TimeNET are components for the analysis of models with a discrete time scale [7, 23]. The model is then referred to as a discrete deterministic and stochastic Petri net (DDSPN). In case of discrete timing, the underlying stochastic process is Markovian if geometrically distributed firing delays are used. The TimeNET component allows immediate transitions as well as transitions with deterministic or geometric firing delay. The geometric distribution corresponds to the exponential in the continuous-time case. TimeNET allows to switch between continuous and discrete time analysis for the same models. Exponential transitions are then interpreted as geometric ones. The advantage of the DDSPN analysis technique is that there can be any number of enabled deterministic transitions in a marking.

Standard numerical methods (Gaussian elimination or SOR) are applied for the steady-state solution of the process. The holding times are then needed to rescale the resulting state probability vector. Normalising the computed result gives the state probabilities of the original model. In case of a transient analysis, the results are computed with an iterative algorithm starting from the initial state.

2.2 Coloured Petri Nets for Manufacturing Systems

Manufacturing systems are a classical application area of Petri nets. The class of coloured stochastic Petri nets used by TimeNET has been introduced especially for the modelling of manufacturing systems, although it is not restricted to it. Two colour types are predefined: *Object tokens* model work pieces inside the manufacturing system, and consist of a name and the current state. *Elementary tokens* cannot be distinguished, and are thus equivalent to tokens from uncoloured Petri nets. Places can contain only tokens of one type. Textual descriptions needed in coloured Petri nets for the definition of variables and colour types can be omitted, and the specification of the types of places and arcs are implicitly given. TimeNET supplies a library of template mod-

els for typical machines and their failure-and-repair behaviour. The models are hierarchically structured, which is necessary to handle complex systems.

Structure and work plans are modelled independently using this net class. This is important for the evaluation of different production plans, where the structural model is not changed. The structural model describes the abilities and work plan independent properties of the manufacturing system resources, such as machines, buffer capacities, and transport connections. Production sequence models specify the work plan dependent features of the manufacturing system. Each route can be thought of as a path through the manufacturing system. Later on, the different model parts are automatically merged resulting in a complete model, which then includes both the resource constraints of the system and the synchronisation of the production steps.

After specifying a manufacturing systems model using the graphical interface, its performance and dependability can be evaluated. Before doing so, the token game can be used to check whether the model correctly describes the system. With TimeNET the modeller is able to either interactively check the model behaviour or to watch an automatic animation of it. However, the main focus is on efficient quantitative evaluation of performance and dependability measures.

Firing delays are associated with transitions for the performance evaluation. We adopt the set of distributions as defined for eDSPNs here. They include immediate, exponential, deterministic, and more general transitions. TimeNET provides components for the steady-state analysis and simulation. The numerical analysis techniques for eDSPNs as described above have been adapted to the coloured net class. This is possible because the stochastic process belongs to the same class.

Details of the component for manufacturing system modelling and evaluation can be found in [24, 25, 26]. There are two additional components which are only briefly mentioned. Users without knowledge of Petri nets can model a manufacturing system with functional blocks originating from the field of production management [27]. Those models are automatically translated into coloured Petri nets and analysed. The provided performance evaluation algorithms can therefore be used without the need to use Petri nets. A recent enhancement is the possibility to directly control a manufacturing system with the token game or animation feature of TimeNET.

3 Tool Usage

For version 3.0 of TimeNET a new generic graphical user interface has been developed. All currently available and future extensions of net classes and their corresponding analysis algorithms are integrated with the same "look-and-feel" for the user. Figure 1 shows a sample screen shot of the interface during a modelling session with coloured Petri nets.

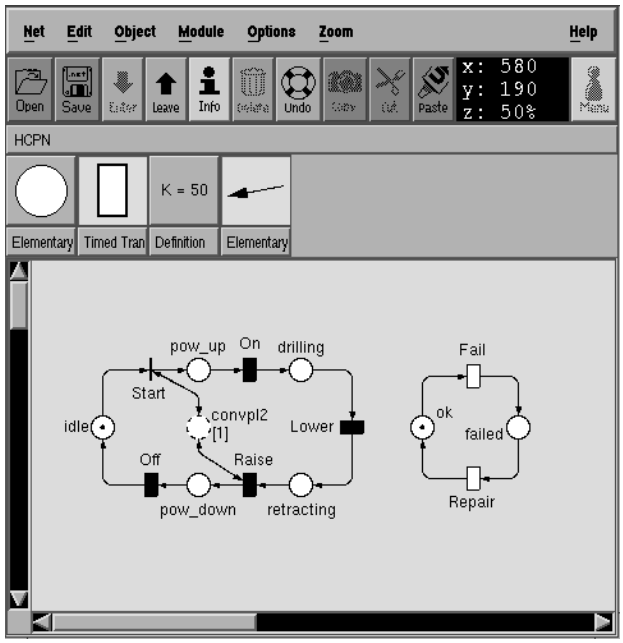

Figure 1: Sample screenshot of the user interface

The upper row of the window contains some menus with basic commands for file handling, editing, and adjusting display options. Under the menu item "Module" the analysis algorithms applicable for coloured Petri nets can be accessed. Some of the most frequently used functions are also available by clicking one of the icons below. Below the icon list the current net class is displayed (HCPN for hierarchically coloured Petri net in this case). The net objects are displayed on buttons in the next row. Each one is shown as an icon and a textual identifier. A group of objects is accessed via each one of these buttons. For instance, all transitions are organised in one group. Clicking on the transition button switches to the next type, but the desired one can also be selected directly by a menu that pulls down from the textual object description.

There are two types of objects available in general,

which form the nodes and arcs of the model class. One node object and one arc object is always selected and highlighted. Textual objects like parameter definitions, performance measures, and comments are special cases of node objects without the possibility to connect arcs with them.

The main drawing area contains a part of the current model. It can be edited like using a standard drawing tool with operations for selecting, moving, and others. Double-clicking on any object displays a window with its attribute identifiers and their current values, e.g. for a place the initial marking and the name. If the object is hierarchically refined (like a substitution transition), double-clicking it displays the refining model.

Depending on the net class, different objects are available in the lower icon list. In addition to that, analysis methods typically are applicable for one net class only. Those methods are integrated in the menu structure of the tool, which therefore changes automatically if a different net class is opened (see subsection 4.1).

It is often necessary to evaluate the system performance for different parameter values in some given range. The corresponding "experiment" feature of TimeNET has been improved in the current version. It is now possible to either linearly or logarithmically change the varying parameter. Experiment descriptions can be stored in a file for later reuse. After the tool has finished all required evaluations for the parameter range, the results are automatically plotted using *gnuplot*.

During the transient analysis and simulation, a window displays the transient development of the specified performance measures. A graphical editor for specifying probability mass functions of transitions with generally distributed firing delays is available.

If a manufacturing system is modelled and evaluated with TimeNET 3.0, several different net classes with their own applicable algorithms are used. The modeller can start with function symbols and translate them into a Petri net with separate models of structure and processing sequences. For this task as well as the modelling of large manufacturing systems with e.g. similar machines the library of Petri net submodels is used. The library in itself is also edited with its own net class in TimeNET. Instantiation of a library model and the translation of function symbol models into Petri nets as well as the compilation of a complete model from the two separate parts are all implemented as net class dependent extensions of TimeNET.

4 Software Architecture of TimeNET 3.0

This section explains some implementation issues of TimeNET 3.0. Solaris on workstations and Linux on personal computers are supported as runtime environments.

4.1 The Generic Graphical User Interface

A generic interface was implemented, that can be used for graph-like models with different types of nodes and arcs. Nodes can be hierarchically refined by corresponding submodels. The interface is called Agnes (**a** **g**eneric **n**et **e**diting **s**ystem) and is not restricted to Petri nets. In fact, it is already being used for other tools than TimeNET. It is implemented in C++ and uses the Motif toolkit.

Two design concepts have been included in the interface to make it applicable for different model classes: A *net class* corresponds to a model type and is defined by a text file. In this file, for each node and arc type of the model the corresponding attributes and the graphical appearance is defined. The shape of each node and arc is defined using a set of primitives (e.g. polyline, ellipse, and text). Shapes can depend on the attribute value of an object, making it possible to show tokens as dots inside places. Figure 1 shows a sample appearance. Nodes can correspond to submodels of a different net class. This facilitates e.g. the organisation of the library submodels like in a file manager.

Functions beyond drawing a model depend on the net class and require programming. Agnes offers the possibility to implement *modules* that are compiled and linked to the program. A module has a predefined interface to the main program. It can select its applicable net classes and extend the menu structure by adding new algorithms. An example of an implemented module applicable without restrictions is an export filter to *xfig*.

With the described user interface, two modellers are able to work together on the same model over the internet.

4.2 Analysis Modules and their Interaction

Figure 2 shows the interaction of the user interface and analysis processes during a direct numerical analysis

of the steady-state behaviour. The overall structure is similar for uncoloured and coloured models. Result data and model description files are stored in readable text files. Most other data is exchanged through Unix sockets to avoid the time-consuming reading and writing of large files on hard disk.

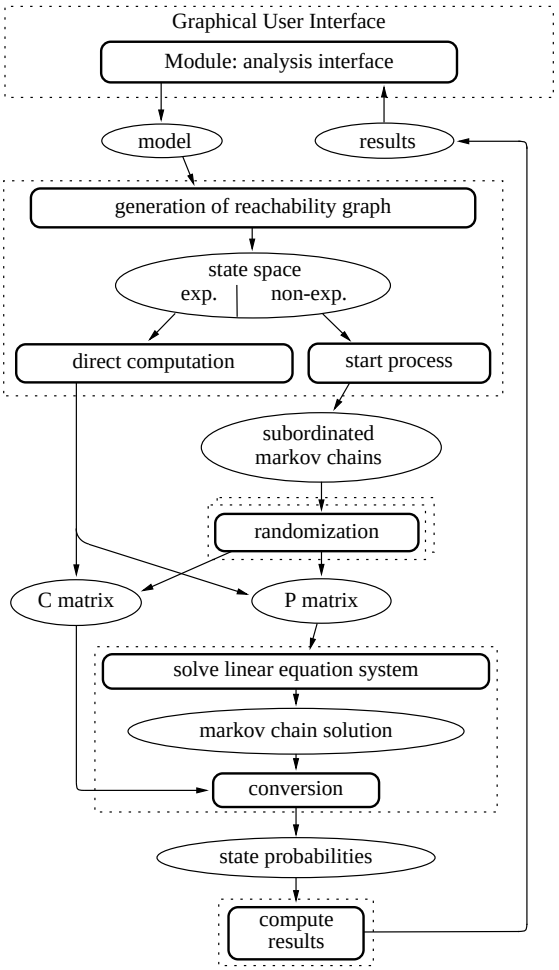

Figure 2: Steady-state numerical analysis processes

By selecting the corresponding menu item, an Agnes module writes the model description to a file and invokes the subsequent processes. The reachability graph is computed in the first step. Techniques as described in [2] are used for a better efficiency, which require a structural analysis of the model.

Afterwards, the one-step probability matrix P and the matrix of mean sojourn times C defining the embedded Markov chain have to be computed. For markings without enabled non-exponential transitions, this can be done directly. Other markings require a transient analysis of a subordinated stochastic processes. The corresponding matrix entries are computed using Jensens method. Furthermore, the algorithm detects isolated components of the subordinated Markov chains as well as isomorphisms between them [20]. Since the components are independent, the randomisation processes can be executed in parallel on different workstations.

The system of linear balance equations given by $\mathbf{P}$ is then solved with standard methods. Successive over relaxation (SOR) and Gaussian elimination are used for this task. The $\mathbf{C}$ matrix is used to convert the result into the time domain, which then has to be normalised. Performance measures can be computed from the resulting vector of steady-state marking probabilities. Similar analysis processes are executed for a model with discrete timing.

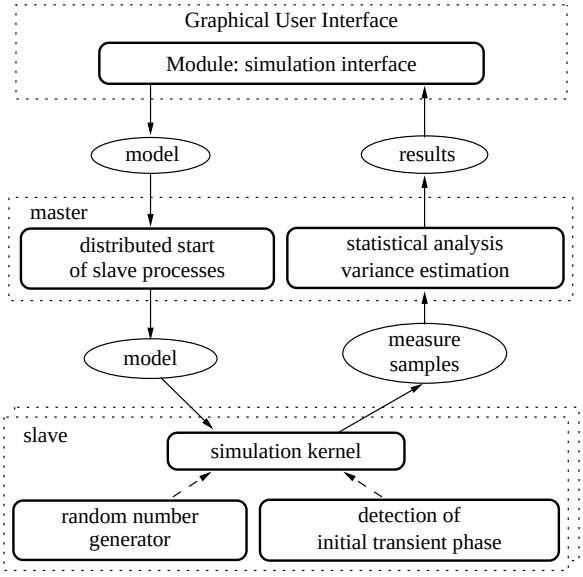

Figure 3: Communication structure of the distributed simulation

For an efficient parallel discrete-event simulation a master/slave concept is used. Figure 3 shows the interaction of the processes. The master starts several replications of slave processes on different workstations. Each slave receives a copy of the whole model and simulates its behaviour. Main parts of the slave are the random number generator as well as an included algorithm for detecting and ignoring the initial transient phase of a simulation run.

Packets of measure samples are sent to the mas-

ter, who uses spectral variance analysis to monitor the already achieved accuracy. The confidence level and maximum relative error of the measures can be set, and the simulation stops automatically after the specified accuracy is reached.

One simulation slave is also running in background during a token game or animation of a coloured Petri net model. The slave is running in a single-step mode and sends information about enabled transitions or state changes after firing one to the graphical user interface.

5 Conclusions

This paper described the software package TimeNET 3.0, which supports the modelling and performability evaluation of discrete event systems. Different classes of Petri net models are available, namely extended deterministic and stochastic Petri nets (eDSPNs) and their coloured extension. Efficient numerical analysis and simulation methods are implemented for the calculation of performance measures either in steady-state or up to a transient point in time. The easily adaptable graphical interface integrates the model specification with the access to analysis modules for all net classes.

The tool is available free of charge for non-profit use. It has been successfully used in many projects and was distributed to more than 150 universities and other organisations worldwide. TimeNET runs under Solaris and Linux.

In the future we plan several extensions of TimeNET: implementation of transient analysis and simulation for coloured models, a new structured modelling method based on Petri nets (Stochastic Petri Net Language [14]), implementation of numerical analysis methods for fluid stochastic Petri nets [22], and methods for efficient performance analysis of large systems [15, 8].

6 Acknowledgements

The development of TimeNET has been partially supported by Siemens AG, Mercedes Benz AG, and numerous grants of the german research council (DFG). The authors would like to thank the former colleagues and numerous students who have contributed to the development and implementation of the software package TimeNET.

References

[1] G. Balbo, G. Chiola, G. Franceschinis, and G. Molinar Roet, "On the efficient construction of the tangible reachability graph of generalized stochastic Petri net models," in *Proc. 2nd Int. Workshop on Petri Nets and Performance Models*, Madison, Wisconsin, 1987, pp. 136–145.

[2] G. Chiola, "Compiling techniques for the analysis of stochastic Petri nets," in *Proc. 4th Int. Conf. on Modeling Techniques and Tools for Performance Evaluation*, Palma de Mallorca, Spain, 1988, pp. 11–24.

[3] G. Chiola, M. Ajmone Marsan, G. Balbo, and G. Conte, "Generalized stochastic Petri nets: A definition at the net level and its implications," *IEEE Transactions on Software Engineering*, vol. 19, no. 2, pp. 89–107, 1993.

[4] G. Chiola, G. Franceschinis, R. Gaeta, and M. Ribaudo, "GreatSPN 1.7: Graphical editor and analyzer for timed and stochastic Petri nets," *Performance Evaluation*, vol. 24, pp. 47–68, 1995.

[5] G. Ciardo, J. Muppala, and K. S. Trivedi, "SPNP: Stochastic Petri net package," in *Proc. 3rd Int. Workshop on Petri Nets and Performance Models*, Kyoto, Japan, 1989, pp. 142–151.

[6] G. Ciardo, R. German, and Ch. Lindemann, "A characterization of the stochastic process underlying a stochastic Petri net," *IEEE Transactions on Software Engineering*, vol. 20, pp. 506–515, 1994.

[7] G. Ciardo and R. Zijal, "Well-defined stochastic Petri nets," in *Proc. 4th Int. Workshop on Modeling, Analysis and Simulation of Computer and Telecommunication Systems (MASCOTS'96)*, San Jose, CA, USA, Apr. 1996, pp. 278–284, IEEE Computer Society Press.

[8] J. Freiheit and A. Zimmermann, "Extending a response time approximation technique to colored stochastic Petri nets," in *Proc. 4th Int. Workshop on Performability Modeling of Computer and Communication Systems (PMCCS)*, Williamsburg, VA, USA, Sept. 1998, pp. 67–71.

[9] R. German, *Analysis of Stochastic Petri Nets with Non-Exponentially Distributed Firing Times*, Dissertation, Technische Universität Berlin, Sept. 1994.

[10] R. German and Ch. Lindemann, "Analysis of stochastic Petri nets by the method of supplementary variables," *Performance Evaluation*, vol. 20, pp. 317–335, 1994.

[11] R. German and J. Mitzlaff, "Transient analysis of deterministic and stochastic Petri nets with TimeNET," in *Proc. 8th Int. Conf. on Computer Performance Evaluation, Modelling Techniques and Tools and MMB (LNCS 977)*, Heidelberg, Germany, 1995, pp. 209–223.

[12] R. German, Ch. Kelling, A. Zimmermann, and G. Hommel, "TimeNET — A toolkit for evaluating non-Markovian stochastic Petri nets," *Performance Evaluation*, vol. 24, pp. 69–87, 1995.

[13] R. German, "New results for the analysis of deterministic and stochastic Petri nets," in *Proc. IEEE Int. Performance and Dependability Symposium*, Erlangen, Germany, 1996, pp. 114–123.

[14] R. German, "SPNL: Processes as language-oriented building blocks of stochastic Petri nets," in *Proc. 9th Conf. Computer Performance Evaluation, Modelling Techniques and Tools*, St. Malo, France, 1997, pp. 123–134.

[15] R. German, "Avoiding fill-in in the analysis of Markov-regenerative stochastic Petri nets," *Proc. 4th Int. Workshop on Performability Modeling of Computer and Communication Systems (PMCCS),* Williamsburg, VA, USA, Sept. 1998, pp. 62–66.

[16] A. Heindl and R. German, "A fourth-order algorithm with automatic stepsize control for the transient analysis of DSPNs," in *Proc. 7th Int. Workshop on Petri Nets and Performance Models*, St. Malo, France, 1997, pp. 60–69 (extended version to appear in IEEE Trans. Softw. Eng.).

[17] Ch. Kelling, "Control variates selection strategies for timed Petri nets," in *Proc. of the European Simulation Symposium*, Istanbul, 1994, pp. 73–77.

[18] Ch. Kelling, "Rare event simulation with RESTART in a Petri net modeling environment," in *Proc. of the European Simulation Symposium*, Erlangen, Germany, 1995, pp. 370–374.

[19] Ch. Lindemann, *Stochastic Modeling using DSPNexpress*, Oldenbourg, 1994.

[20] Ch. Lindemann, "Exploiting isomorphisms and special structures in the analysis of Markov regenerative stochastic Petri nets," in W. J. Stewart (Ed.) *Numerical Computation with Markov Chains*, pp. 383–402, Kluwer, 1995.

[21] W. H. Sanders, W. S. Obal II, M. A. Qureshi, and F. K. Widjanarko, "The UltraSAN modeling environment," *Performance Evaluation*, vol. 24, 1995, pp. 89–115.

[22] K. Wolter, "Second order fluid stochastic Petri nets: an extension of GSPNs for approximate and continuous modelling," in *Proc. World Congress on Systems Simulation*, Singapore, Sept. 1997, pp. 328–332.

[23] R. Zijal, *Analysis of Discrete Time Deterministic and Stochastic Petri Nets*, Dissertation, Technische Universität Berlin, Oct. 1997.

[24] A. Zimmermann, S. Bode, and G. Hommel, "Performance and dependability evaluation of manufacturing systems using Petri nets," in *1st Workshop on Manufacturing Systems and Petri Nets, 17th Int. Conf. on Application and Theory of Petri Nets*, Osaka, Japan, 1996, pp. 235–250.

[25] A. Zimmermann, *Modellierung und Bewertung von Fertigungssystemen mit Petri-Netzen*, Dissertation, Technische Universität Berlin, Sept. 1997, (in german).

[26] A. Zimmermann and J. Freiheit, "TimeNET$_{\text{MS}}$ — An integrated modeling and performance evaluation tool for manufacturing systems," in *IEEE Int. Conf. on Systems, Man, and Cybernetics*, San Diego, USA, 1998, pp. 535–540.

[27] A. Zimmermann, A. Kühnel, and G. Hommel, "A modelling and analysis method for manufacturing systems based on Petri nets," in *Computational Engineering in Systems Applications (CESA '98)*, Nabeul-Hammamet, Tunisia, 1998, pp. 276–281.

[28] A. Zimmermann and G. Hommel, "Modelling and evaluation of manufacturing systems using dedicated Petri nets," *Int. Journal of Advanced Manufacturing Technology*, vol. 15, 1999, pp. 132–137.