

Parallele (ereignisorientierte) Simulation  
von stochastischen Petri-Netzen

# **PES for SPN**

— Vorschlag für eine Diplomarbeit —

Armin M. Warda

(warda@ls4.informatik.uni-dortmund.de)

3. Mai 1995

## **Zusammenfassung**

Dieses Dokument stellt eine erste provisorische Zusammenstellung von Ideen und Konzepten für eine Diplomarbeit am Lehrstuhl Informatik IV der Universität Dortmund dar.

Ziel der Diplomarbeit ist es, Konzepte für die simulative, quantitative Analyse von stochastischen Petri-Netzen auf Parallelrechnern und/oder Rechnerverbunden darzustellen, zu bewerten und vergleichen sowie eine geeignete Methode prototypisch im Rahmen des in Weiterentwicklung befindlichen Software-Tools QPN zu implementieren.

Ein wesentlicher Schwerpunkt der Diplomarbeit sollten Methoden zur Partitionierung der Modelle bzw. ihrer Simulation sein.

## **Inhaltsverzeichnis**

<b>1</b>	<b>Ein paralleler Simulator für QPN</b>	<b>3</b>
<b>2</b>	<b>Logisches Schichtenmodell des Simulators</b>	<b>4</b>
<b>3</b>	<b>Themenbereiche</b>	<b>6</b>
<b>4</b>	<b>Betreuung</b>	<b>7</b>
<b>5</b>	<b>Zeitraumen</b>	<b>7</b>
	<b>Literatur</b>	<b>7</b>



# 1 Ein paralleler Simulator für QPN

Derzeit sind in das Software-Tool QPN [Kemp91, Bau93a, Bau93b, BaKe93] verschiedene numerisch-analytische Löser zur qualitativen und quantitativen Analyse von stochastischen Petri-Netzen integriert. Obwohl mit diesen Methoden eine große Modell-Klasse behandelbar ist, besteht doch bei anderen Modellen, z.B. mit sehr großem Zustandsraum, die Hoffnung, diese durch Simulation effizienter zu lösen oder einer Lösung überhaupt erst zugänglich zu machen.

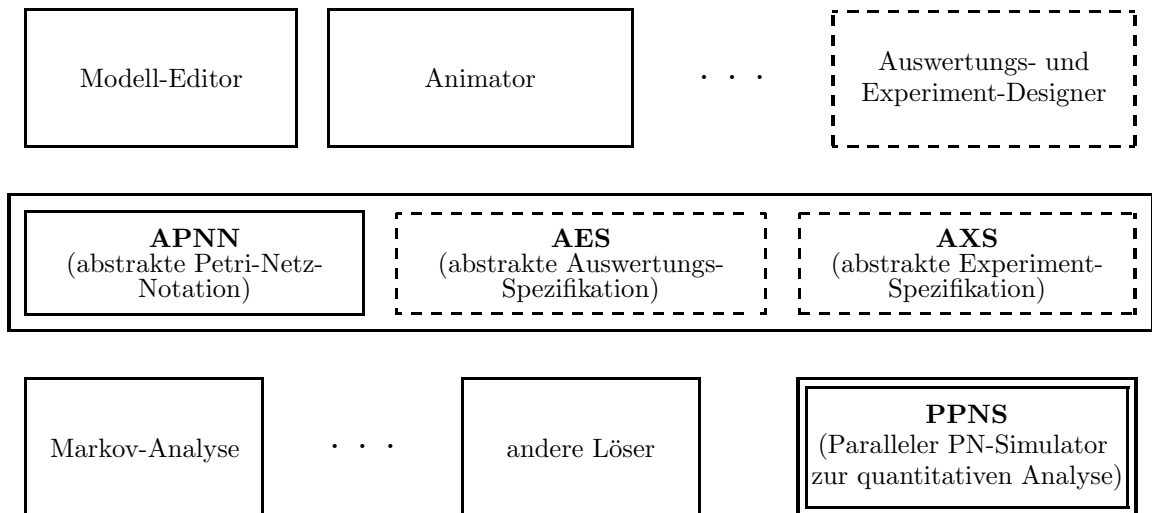


Abbildung 1: Einordnung des Simulators in das Konzept von QPN

QPN besteht aus einer Anzahl von unabhängigen Modulen. Der Datenaustausch zwischen den einzelnen Modulen, etwa einem Modelleditor und einem Löser erfolgt über eine wohldefinierte Schnittstelle, die *Abstract Petri Net Notation (APNN)* [?]. Diese Schnittstelle, die weitestgehend unabhängig von speziellen Lösungsmethoden und Modellklassen ist, soll auch der Ansatzpunkt für die Integration des Simulators, sein.

Die Implementierung des Simulators kann also unabhängig von der graphischen Benutzeroberfläche und anderen Lösern erfolgen, es müssen lediglich noch geeignete Wege gefunden werden, die Spezifikationen für die Modellauswertung und Experimente dem Löser mitzuteilen und schließlich die ermittelten Ergebnisse zurückzuführen. Bei letzterem ist eine Dateiausgabe in Tabellenform zumindest in einer prototypischen Implementierung sicher zunächst befriedigend, für die Spezifikation von Modellauswertungen und Experimenten wäre aber eine (in Anlehnung an die APNN) abstrakte, wohldefinierte Schnittstelle sinnvoll.

## 2 Logisches Schichtenmodell des Simulators

Bei dem Entwurf eines parallelen ereignisorientierten Simulators steht zunächst die Entscheidung an, einen konservativen [Misra86] oder optimistischen Ansatz [Jeff85] zu wählen. Fallstudien, wie etwa [FeCh94], zeigen, daß für die parallele Simulation von bestimmten Klassen von Petri-Netzen optimistische Ansätze gut geeignet sind. Es besteht somit Anlaß zur Hoffnung, daß sich diese Methoden auch für *stochastische* Petri-Netze lohnen. Weitere Argumente für optimistische Ansätze sind das elegante Konzept *Virtual Time* [Jeff85] sowie die sympathische Eigenschaft der *Deadlock-Freiheit* im Gegensatz zu konservativen Methoden.

Abbildung 2 veranschaulicht eine mögliche Strukturierung des parallelen Petri-Netz-Simulators in Schichten mit bestimmter Funktionalität und (von unten nach oben) wachsender Abstraktion von der realen Maschine. Jede dieser Schichten realisiert ein bestimmtes Prozeß-Modell: während die reale Maschine (Hardware plus Betriebssystem) beispielsweise nur asynchrone parallele Berechnungen mit expliziter Synchronisation durch Nachrichtenaustausch oder Kommunikation über gemeinsamen Speicher anbietet, kann die aus den unteren drei Schichten bestehende virtuelle Maschine synchrone parallele Berechnungen durchführen.

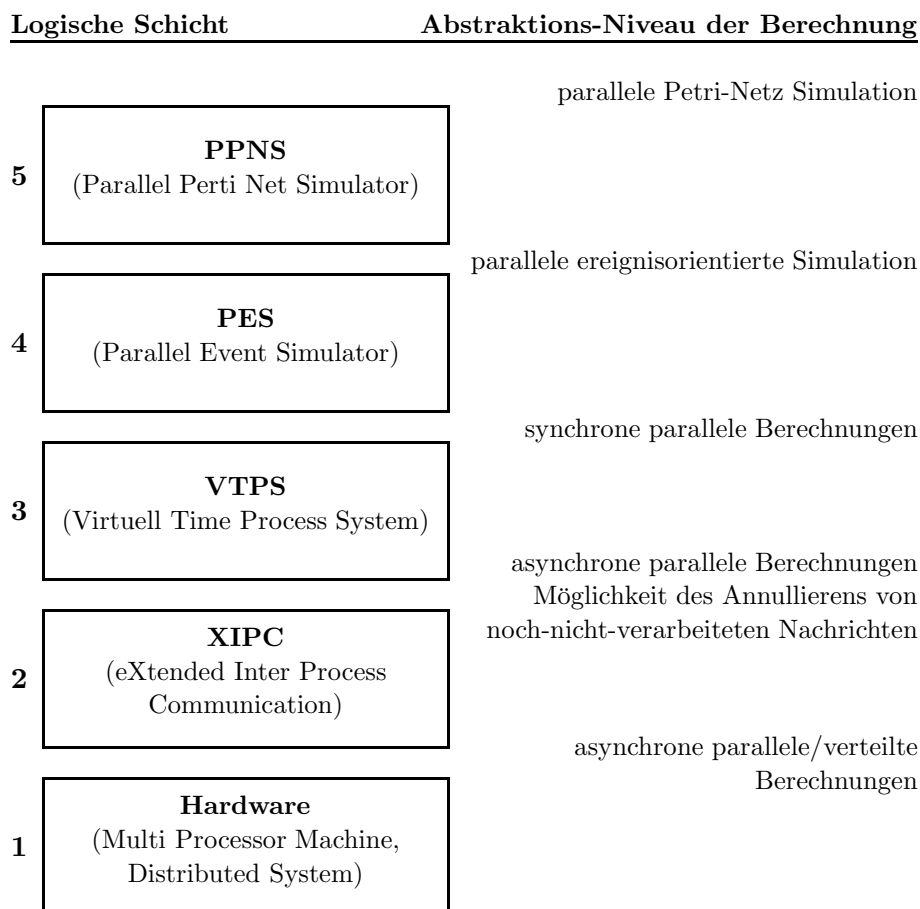


Abbildung 2: Logische Schichten des Simulators

Im folgenden wird die Funktionalität der einzelnen Schichten skizziert:

1. **Hardware** (Multi Processor Machine, Distributed System)  
 Parallele Prozesse laufen asynchron, können sich durch Nachrichten oder gemeinsame Speicherbereiche synchronisieren.
  
2. **XIPC** (eXtended Inter Process Communication)  
 Abstrahiert von der „Struktur“ der Hardware (eng gekoppelt, z.B. Multi Processor Machine – gegenüber lose gekoppelt, z.B. Rechner im Netz).  
 Realisiert einen (nicht notwendigerweise reihenfolgetreuen) Datagramm-Dienst mit der Möglichkeit, verschickte - aber noch nicht vom Empfänger verarbeitete - Nachrichten durch Senden von *Antimessages* zu annullieren (**un\_send**). Des weiteren kann eine empfangene Nachricht wieder in den Empfangspuffer zurückgelegt werden (**un\_receive**), sodaß sie anschließend noch einmal empfangen werden kann.
  
3. **VTPS** (Virtuell Time Process System)  
 Realisiert Prozesse mit lokaler virtueller Zeit und Verwaltung von Zustandsraum-Versionen mit Operationen zum Sichern des aktuellen Zustands, Löschen von veralteten Zustandsinformationen und Rücksetzen auf alte Zustände. Dies ist im wesentlichen die Funktionalität des *Roll-Back-Chips* [Fuji90]. Der verteilte Rollback wird durch *Antimessages* implementiert.  
 Insgesamt realisiert diese Schicht virtuell-synchrone Prozesse entsprechend dem *Time-Warp*-Mechanismus [Jeff85, Jeff87].
  
4. **PES** (Parallel Event Simulator)  
 Realisierung von allgemeinen Konzepten der ereignisorientierten Simulation und globalen Aspekten des *Time-Warp*-Algorithmus. Das Konzept von *Antimessages* ist in dieser Schicht nicht mehr sichtbar.
  
5. **PPNS** (Parallel Petri Net Simulator)  
 Bildet Petri-Netz-Modelle (zusammen mit Spezifikationen für Auswertung und Experimente) auf die parallele ereignisorientierte Simulation ab. Dazu gehört insbesondere die Partitionierung.  
 Aspekte der Hardware-Struktur müssen hier (in einer geeigneten abstrakten Weise) sichtbar sein, damit die Partitionierung auf unterschiedliche Prozessorleistungen und Kommunikationskosten Rücksicht nehmen kann.

Neben dem Vorteil der Modularisierung und der Chance, durch den wachsenden Abstraktionsgrad der Schichten die Komplexität der gestellten Aufgabe zu meistern, hat dieses Modell noch weitere sympathische Eigenschaften: oberhalb der 2. Schicht ist weitgehende *Hardware-Unabhängigkeit* gegeben, sieht man einmal davon ab, daß bei der Partitionierung natürlich bestimmte Leistungsmerkmale der Hardware zu berücksichtigen sind. Die Schichten 1 bis 4 sind andererseits *Anwendungs-unabhängig*, d.h. prinzipiell ließen sich darauf aufbauend auch Simulationen anderer Modellwelten realisieren.

### 3 Themenbereiche

Die folgende (unvollständige, im wesentlichen unsortierte) Liste enthält Themenbereiche, die für die Diplomarbeit relevant sind. Daraus ergeben sich vielfältige Möglichkeiten der Schwerpunktbildung.

- Vorstellung: PES, konservativ vs. optimistisch, Virtual Time, Time-Warp
- Vorstellung: SPNs und spezielle Klassen
- Probleme bei der ereignisorientierten Simulation von PN
  - gleichzeitige Ereignisse (immediate Transition)
  - ...
- Untersuchung, Vergleich und Bewertung der Time-Warp-Varianten
- Testmenge von Modellen zum „Benchmarking“ des Simulators
- Ergänzung der APNN um abstrakte Spezifikationen für Modellauswertung und Experimentdurchführung
- Erweiterung der APNN um Modell-Elemente, die analytisch-numerisch nicht — jedoch simulativ problemlos — behandelbar sind, etwa
  - beliebige Verteilungen für Zufallsvariablen
  - unbeschränkte Populationen
  - ...
- Partitionierung der Simulation gemäß
  - Struktur: Subsysteme, Hierarchien?
  - Struktur: schwach-zusammenhängende Teilnetze?
  - Modellverhalten: nach einer (numerischen) Grobanalyse?
  - ...
- dynamische Anpassung der Partitionierung: Load-Balancing
- transiente Analyse
- Überprüfung von stabilen Prädikaten  $\exists t_0 \forall t \geq 0 : f(t_0) = \text{wahr} \Rightarrow f(t_0 + t) = \text{wahr}$
- überhaupt sinnvoll, verteilt zu simulieren, wenn Nachrichtenlaufzeiten so groß wie in Netzen?
- Verbesserung durch Broadcasting/Multicasting/Nachrichtenbündelung?
- man könnte aber auch die automatische Verteilung von replizierten Läufen und Experimentserien auf die Rechner eines Netzes unterstützen!
- ...

## 4 Betreuung

Für die Betreuung der Diplomarbeit bieten sich Falko Bause und Peter Kemper an.

## 5 Zeitrahmen

	Nov.-Dez.	1994	Prüfung im Vertiefungsgebiet: Simulation und SPN
-3.5	Jan.	1995	<u>Beginn Einarbeitung in QPN</u>
	Jan.-Apr.	1995	Prüfung Nebenfach Mathematik Prüfung Inf. I (Theorie) Theoretische Vorarbeiten
0	Apr.-Mai	1995	<u>Anmeldung Diplomarbeit</u> Implementierung Prototyp
+2	Juni-Juli	1995	<u>Fertigstellung Prototyp Simulator (manuelle Partitionierung)</u> Experimentieren, Benchmarking mit Prototyp Implementierung Partitionierung
+4	Aug.-Sep.	1995	<u>Fertigstellung Prototyp Partitionierung</u> Experimentieren, Benchmarking mit Prototyp
+6	Okt.-Nov.	1995	<u>Fertigstellung Diplomarbeit</u>
	Ende	1995	Abschluß des Studiums: Diplom

## Literatur

- [Ajmo93] Ajmone Marsan, Marco [Hrsg.]: *Application and Theory of Petri Nets*; Proceedings of the 14. Int. Conf. on Applications and Theory of Petri Nets, Chicago, Ill. Springer, Berlin, 1993.
- [Bau93a] Bause, F.: *Queueing Petri nets: A Formalism for the Combined Qualitative and Quantitative Analysis of Systems*; 5. Int. Workshop on Petri Nets and Performance Models, 20.-22. October 1993, Toulouse, France.
- [Bau93b] Bause, F.: “ $QN + PN = QPN$ “ — *combining Queueing Networks and Petri Nets*; Forschungsberichte des Fachbereichs Informatik der Universität Dortmund, 461, 1993.
- [Bau94] Bause, F.: *Combining qualitative and quantitative analysis of generalized stochastic Petri nets*; Forschungsberichte des Fachbereichs Informatik der Universität Dortmund, 527, 1994.
- [BaBe90] Bause, F.; Beilner, H.: *Eine Modellwelt zur Integration von Warteschlangen- und Petri-Netz-Modellen*; 5. GI/ITG- Fachtagung Messung, Modellierung und Bewertung von Rechensystemen, Braunschweig, 1989, Informatik-Fachbericht 218, Springer-Verlag.

- [BaKe93] Bause, F.; Kemper, P.: *Queueing Petri Nets*; 3. Fachtagung Entwurf Komplexer Automatisierungssysteme 1993, TU Braunschweig.
- [BaKr93] Bause, F.; Kritzinger, P.: *Introduction to Stochastic Petri Net Theory*; Dept. of Computer Science, University of Cape Town, South Africa, May 1993.
- [FeCh94] Ferscha, A.; Chiola, G.: *Accelerating the Evaluation of Parallel Program Performance Models Using Distributed Simulation*; Proc. 7th Int. Conf. on Computer Performance Evaluation, Springer-Verlag, 231-252, Mai 1994.
- [Fuji90] Fujimoto, R.M.: *Parallel Discrete-Event Simulation*; Communications of the ACM, 33(10): 30-53, Oktober 1990.
- [FuTs92] Fujimoto, R.M.; Tsai, J.J.: *Design and Evaluation of the Rollback Chip: Special Purpose Hardware for Time-Warp*; IEEE Transactions on Computers, 41(1): 68-81, Januar 1992.
- [Jeff85] Jefferson, D.R.: *Virtual time*; ACM Transactions on Programming Languages and Systems, 7(3): 404-425, Juli 1985.
- [Jeff87] Jefferson, D.R. et al.: *Distributed Simulation and the Time-Warp Operating System*; Proc. 12th SIGOPS-Symposium on Operating Systems Principles, 77-93, 1987.
- [Kemp91] Kemper, P.: *Untersuchung einer kombinierten Warteschlangen-Petrinetz-Modellwelt*; Diplomarbeit, Universität Dortmund, Informatik IV, 1991.
- [LaKe82] Law, A. M.; Kelton, D.: *Simulation Modelling and Analysis*; McGraw-Hill, London, 1982.
- [Misra86] Misra, J.: *Distributed Discrete-Event Simulation*; ACM Computing Surveys, 18(1): 39-65, März 1986.
- [Paul86] Paul, W.: *Ein Verfahren zur effizienten Simulation modifizierter Petri-Netze*; Diplomarbeit, Universität Dortmund, 1986.
- [SAD86] Singh, M. G.; Allidina, A. Y.; Daniels, B. K.: *Parallel processing techniques for simulation*; Plenum, New York, 1986.