QueueSimulation: Eine Simulationsumgebung zum Modellieren und Studieren von Wartesystemen

Andreas Rinkel

Abteilung Informatik
Hochschule Rapperswil,
Institut für Internet-Technologien und –Anwendungen,
CH-8640 Rapperswil
andreas.rinkel@hsr.ch

Abstract: Warteschlangensysteme bilden in der Informatikausbildung eine wichtige Grundlage zur Leistungsbewertung von Systemen. Leider ist der analytische Zugang für die eher praktisch ausgerichteten Studiengänge, z.B. der Fachhochschulen nicht passend. In dem folgenden Aufsatz wird ein Ansatz gezeigt, wie das abstrakte Thema Warteschlangen auch ohne den üblichen mathematischen „Ballast“ vermittelt werden kann. Ferner können mit der an der HSR entwickelten Simulationsumgebung *QueueSim* neben der Überprüfung errechneter Werte auch praxisrelevante Problemstellungen bearbeitet und untersucht werden.

1 Motivation

In der Informatikgrundlagenausbildung bildet die Theorie der Warteschlangen [KL76] einen wesentlichen Bestandteil. Benötigt werden die vermittelten Kenntnisse zur Leistungsbewertung von z.B. Computernetzen, verteilten Applikationen oder allgemein von Wartesystemen [Gu99]. Leider erfreuen sich diese Grundlagenveranstaltungen wegen ihres hohen Abstraktionsgrades, dem erforderlichen mathematischen Apparat und der scheinbar geringen Praxistauglichkeit bei Studenten und Praktikern einer eher geringen Popularität. Um dennoch in praxisbezogenen Ausbildungszweigen, wie z.B. an Fachhochschulen, die erforderlichen Grundlagen erfolgreich zu vermitteln, sind einerseits die mathematischen Anforderungen auf das Nötigste zu beschränken und andererseits die Praxistauglichkeit durch anschauliche Übungen und Beispiele zu untermauern. Im Rahmen der Vorlesung Computernetze an der Fachhochschule Rapperswil, Schweiz, entstand ein Modul zur Einführung in die Leistungsbewertung, das diesen besonderen Bedingungen Rechnung trägt. Der grundsätzliche Ansatz der Veranstaltung wird im Abschnitt Vorlesungskonzept dargestellt.

Zur praktischen Anwendung wurde an der HSR das Simulationssystem *QueueSim* im Rahmen von zwei Studienarbeiten entwickelt. Das in Abschnitt 3 vorgestellte Simulationssystem erlaubt den StudentInnen über eine fensterorientierte Oberfläche mittels *drag and drop* Simulationsmodelle zu erstellen und zu studieren. Der Bericht schliesst mit ersten Erfahrungen und einem Ausblick auf weitere Projekte.

2 Vorlesungskonzept

In der Vorlesung bzw. dem Unterricht werden ausgehend von einer eingehenden Problemanalyse einer *Campus*-Netzwerkarchitektur [Ch00] typischer Fragen des Leistungsverhaltens formuliert, z.B:

- Werden Anfragen an den Server XY in 85% der Fälle in maximal 6 Sekunden beantwortet?
- Welchen Einfluss hat es auf die Netzauslastung, wenn weitere 50 Benutzer auf einen bestimmten Server zugreifen?
- Wie verändert sich die Netzauslastung, wenn statt einer Haustelefonanlage eine *Voice over* IP Lösung angestrebt wird?

Anschliessend werden die Modellierungsschritte vom realen System hin zum Verkehrsmodell bestehend aus Ankunftsprozess und Bedienprozess verdeutlicht. Als Beispiel dient der Datenverkehr, den ein Router zu bearbeiten hat (Abbildung 1). In einem ersten Schritt wird das Verhalten eines einzelnen Benutzers beschrieben. Durch das Benutzerverhalten, z.B. E-Mail-Verkehr, Surfverhalten und Serverzugriffe, lässt sich das daraus resultierende, erzeugte mittlere Datenvolumen pro Zeiteinheit ermitteln. Dieser, durch einen einzigen Benutzer erzeugte Verkehr, kann durch eine Zufallsvariable λ und eine Verteilungsfunktion abstrahiert beschrieben werden. Der Gesamtverkehr einer Benutzergruppe wird dann durch einfaches aufsummieren der Teilverkehre λ_i gebildet. Die Summe der Teilverkehre beschreiben den Ankunftsprozess.

In einem weiteren Abstraktionsschritt ist das Verhalten des Routers aus Sicht des Durchsatzverhaltens zu beschreiben.

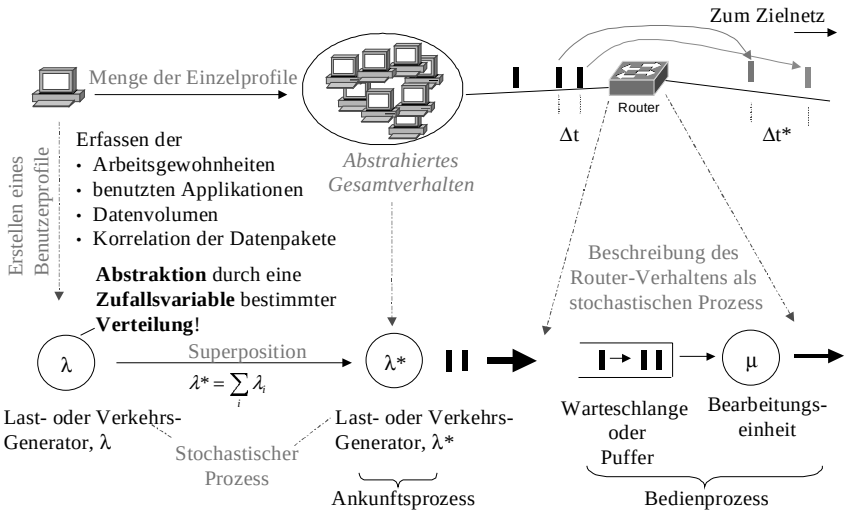

Abbildung 1: Modell und Ableitung der Kenngrössen λ und μ

Leicht zu beobachten ist der Datenstrom einzelner Datenpakte oder Token zum Router hin und vom Router weiter in das Zielsubnetz. Wird nur das zeitliche Verhalten betrachtet, so kann der Router als Verzögerungsglied mit Zwischenspeicher aufgefasst werden, der eine mittlere Anzahl von Datenpaketen zwischenspeichert und diese nach einer mittleren Bearbeitungszeit in das Zielsubnetz weiterleitet. Hierbei ist zu beobachten, dass die Bearbeitungszeiten unterschiedlich sind ($\Delta t \neq \Delta t^*$) und einer bestimmten Verteilung unterliegen. Allgemein gilt: Token werden bei Bedarf innerhalb des Routers zwischengespeichert (→ Warteschlange) und abhängig von Netzsituation und Paketgrösse nach einer mittleren Bearbeitungszeit s (→ Bearbeitungseinheit) weitergeleitet. Die durchschnittliche Bearbeitungs- oder Servicezeit s unterliegt ebenfalls einer Verteilungsfunktion mit dem Mittelwert $s = 1/\mu$. Für die Verteilungsfunktionen des Ankunfts- bzw. des Bedienprozesses mit den Kenngrössen μ und λ wird zunächst stillschweigend die negativ exponentielle Verteilungsfunktion angenommen. Die negativ exponentielle Verteilungsfunktion zeichnet sich dadurch aus, dass sie erinnerungsfrei ist, d.h. die Historie bereits aufgetretener Token keinen Einfluss auf das Auftreten neuer Token besitzt. Unter dieser a priori Annahme können nun ohne weiteren „Ballast“ die Berechungsgrundlagen für das Warteschlangen-Grundmodell im stabilen Zustand (die M/M/1 Queue, siehe Abbildung 2)angegeben werden.

Stabil ist das System dann, wenn im Mittel genau so viele Token in das System eintreten, wie das System verlassen, d.h. $\mu > \lambda$ ist. Die Gesamtaufenthaltszeit R ergibt sich aus der Wartezeit W plus der Servicezeit s. Mit Hilfe von Little’s Gesetz lässt sich dann die mittlere Gesamtaufenthaltszeit R als Funktion der Servicezeit s und der Ankunftsrate λ bestimmen.

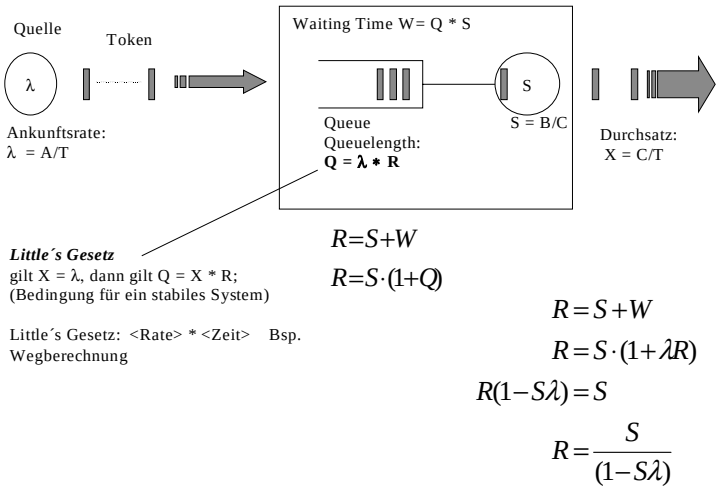

Abbildung 2: Berechnung der M/M/1-Queue

Ausgehend vom Grundmodell der M/M/1 Queue können weiterer Modelle, wie z.B. Dual-Server oder Feedback-Center hergeleitet werden.

Im Übungsbetrieb sind von den Studierenden zunächst die typischen Aufgaben am Beispiel stark vereinfachter Systeme zu lösen, die sich lediglich auf die Grundmodelle beschränken. Dabei sollen die Studierenden erst ein Gefühl für den vorgestellten Modellansatz und dessen Möglichkeiten entwickeln. Im weiteren Verlauf sind jedoch auch praxisrelevante System zu modellieren und zu studieren. Da der analytische Ansatz eines komplexen Systems mit beliebigen Verteilfunktionen für die Ankunfts- und Bearbeitungsprozess einen erheblichen mathematischen Aufwand darstellt, wurde für den praktischen Gebrauch der Simulator *QueueSim* entwickelt. Die Simulation erlaubt es den Studierenden in einfacher Weise einerseits den bisher erlernten Stoff praktisch nachzuvollziehen und andererseits weitere komplexe Systeme zu modellieren und zu studieren. Besonders interessant ist hier, dass die Simulation nicht nur den stationären Zustand ermittelt sondern auch das Einschwingverhalten untersucht werden kann.

3 Die Simulationsumgebung *QueueSim*

3.1 Aufbau

Zur Abbildung der Warteschlangensysteme in ein Simulationsmodell [FL79] konnte auf die an der HSR entwickelte Java Klassenbibliothek *J-TOOPS* (Java Tools for Object Oriented Process Simulation) aufgebaut werden. Die Klassenbibliothek erlaubt:

- die Beschreibung nebenläufiger und konkurrierender Prozesse, bzw. Objekte,

- die Festlegung logischen und temporalen Verhaltens in beliebiger Granularität,

- die Definition komplexer Kommunikationsstrukturen zwischen Objekten

und wurde bisher hauptsächlich im Forschungsbereich, z.B. zur Simulation des UMTS Powermanagements [MKR02] eingesetzt.

Das Verhalten der Wartesysteme und der Fluss der Token lässt sich leicht auf die Elemente der Klassenbibliothek abbilden. So lassen sich Modellserver als parallele Prozesse darstellen, die Token entgegennehmen, ggf. zwischenspeichern und nach einer einstellbaren mittleren Wartezeit wieder an die nächste Warteschlange weitergeben. Hierbei ist die Einstellung der Verteilungsfunktion wählbar. Die Modellierung der Modell-Lastgeneratoren wird ebenfalls auf parallele Prozesse abgebildet. Zur sicheren Wiederholung der Experimente werden zur Beschreibung der Zufallsprozesse Pseudo-Zufallszahlen erzeugt.

Die Implementierung der fenstergesteuerten Simulationsumgebung *QueueSim* wurde im Rahmen zweier paralleler Studienarbeiten [ZT03], [MS03] vollständig in Java durchgeführt.

3.2 Das Simulationstool QueueSim

Nach Start der Applikation *QueueSim* öffnet sich das Hauptfenster (Abbildung 3). Von hier lassen sich über die Menüleiste weitere Arbeitsbereiche öffnen. Zur Verwaltung von Simulationsläufen werden diese in sogenannten Projekten (Abbildung 3, Punkt 1) abgelegt. Innerhalb eines Projektes können mehrere Simulationsmodelle und deren Ergebnisse verwaltet werden. So können auch Vergleichsmessungen zwischen verschiedenen Simulationen eingestellt werden, so dass der Modellierer die Simulationsergebnisse verschiedener Modellalternativen leicht vergleichen kann. Im Arbeitsbereich wird das eigentliche Simulationsmodell erstellt (Abbildung 3, Punkt 2). Hierzu stehen den ModelliererInnen verschiedene Modellgrundkomponenten zur Verfügung, die mit *drag and drop* von der Funktionsbibliothek (Abbildung 3, Punkt 3) in den Arbeitsbereich gezogen und verbunden werden können. Neben dem direkten Verbinden zweier Grundkomponenten können Verkehrsströme auch zusammengefasst oder aufgeteilt werden. Hierzu stehen den ModelliererInnen die Komponenten *Merger* und *Splitter* der Funktionsbibliothek zur Verfügung. So lassen sich auch Rückkopplungen einfach modellieren.

Die Eigenschaften der einzelnen Modellkomponenten werden über den Eigenschaftseditor (Abbildung 3 Punkt 4) festgelegt. Der Eigenschaftseditor kann zu jeder Modellkomponente aufgerufen werden. Die Studierenden können auch während der Simulation die Eigenschaften einer Komponente, wie z.B. die mittlere Servicezeit verändern, um so interaktiv am Modell zu experimentieren und Modellveränderungen unmittelbar im Messreport oder Plot anzuschauen. Der Simulationsablauf wird über die Simulationssteuerung (Abbildung 3 Punkt 5) geregelt. Hier kann neben den Simulationsstart, -stopp oder -pause Befehlen zwischen einer animierten oder nicht-animierten Simulation entschieden werden. Im Falle der animierten Simulation ist die Geschwindigkeit der Animation in bestimmten Grenzen einstellbar.

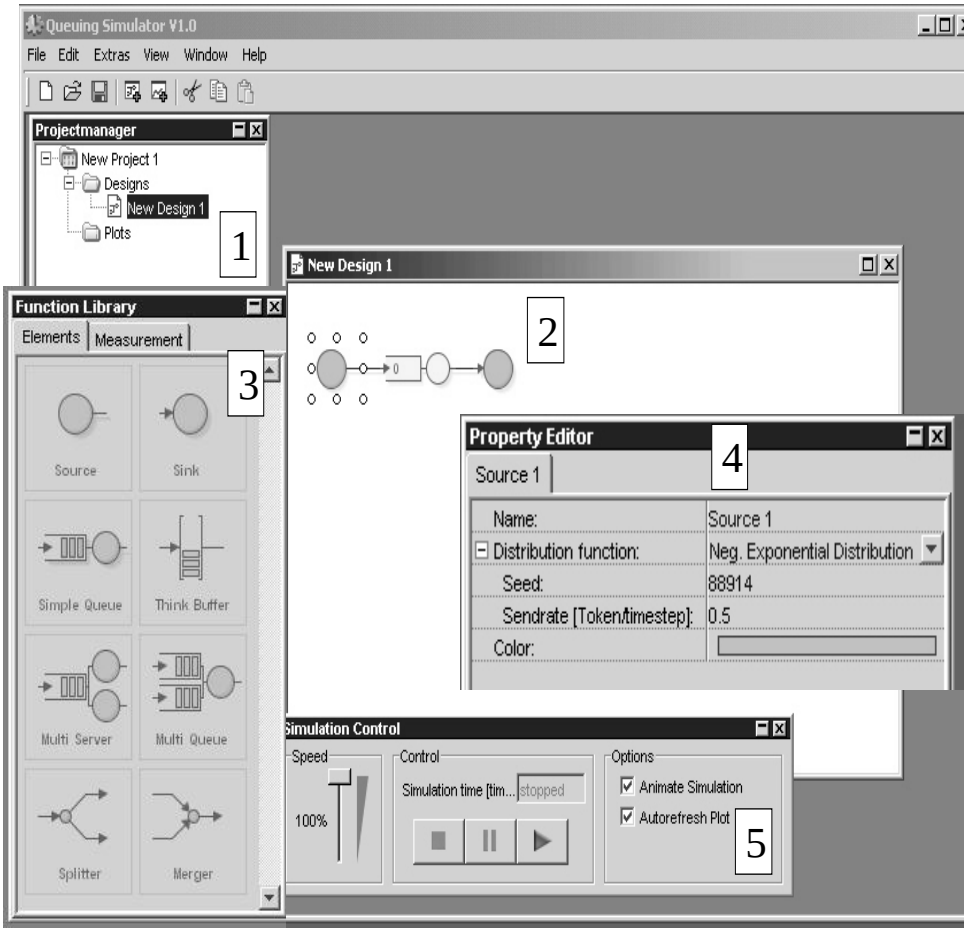

Abbildung 3: Hauptfenster der Simulationsumgebung QueueSim

3.3 QueueSim am Beispiel der M/M/1 Warteschlange

Am Beispiel der M/M/1 Warteschlange soll der Einsatz des Simulators illustriert werden. Wie aufgezeigt, ermittelt sich die durchschnittliche Aufenthaltsdauer R eines *Tokens* innerhalb eines M/M/1 Systems zu R = S/(1-λS). Mit einer mittleren Ankunftsrate von λ=0.5s und einer mittleren Bearbeitungszeit von s = 1s ergibt sich daraus eine mittlere Aufenthaltsdauer eines Tokens von R=2s.

Mit Hilfe des Simulationssystems *QueueSim* kann das Ergebnis leicht simulativ ermittelt werden. Abbildung 4 zeigt auf der linken Seite den Arbeitsbereich der Simulationsumgebung. Das Simulationsmodell besteht aus einer *Quelle* mit der gesetzten Variablen λ = 0.5 Token pro Zeiteinheit. Der erzeugte Verkehr wird an die Komponente *SimpleQueue* mit der eingestellten mittleren Bedienzeit von 1 Zeiteinheit pro *Token* weitergeleitet. Schliesslich verlassen die *Token* die einfache Warteschlange und werden in der Senke aufgenommen. Die Senke hat keine besondere Funktion, sondern sie dient nur zum definitiven Abschluss des *Tokenflusses.* Mit der oben ermittelten Formel müsste sich im eingeschwungenen oder stationären Zustand eine mittlere Aufenthaltsdauer von R = 2 Zeiteinheiten einstellen. Der auf der rechten Seite dargestellte Messplot zeigt die zeitliche Entwicklung des Systems über der Simulationszeit. Nach dem Einschwingvorgang stellt sich die mittlere Aufenthaltszeit R auf den vorher berechneten Wert von 2 Zeiteinheiten ein.

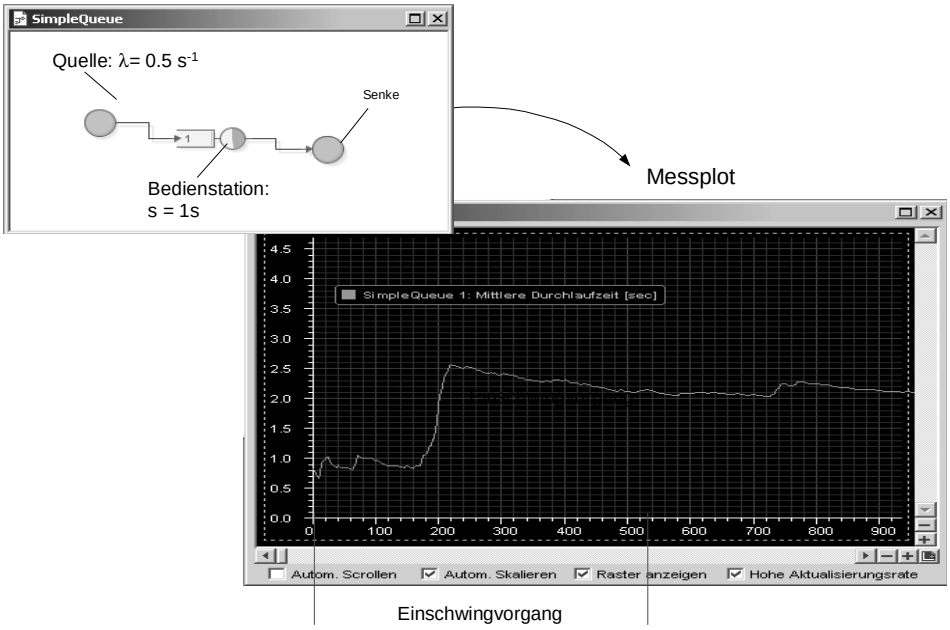

Abbildung 4: Modell der M/1/1 Simulation

Wird hingegen der Systemparameter für die Bearbeitungszeit auf s=2.5s erhöht, so befindet sich das System jenseits der Stabilitätsgrenze und die Aufenthaltszeit R steigt über alle Grenzen an. Dieser Fall ist in dem Simulationsergebnis Abbildung 5 dargestellt. In der Animation der Simulation wird die instabile Warteschlange rot eingefärbt und der Zeitpunkt, wann die Instabilität vom System entdeckt wurde, angezeigt. Im Plot ist der Anstieg der nicht konvergierenden Aufenthaltzeit R zu erkennen.

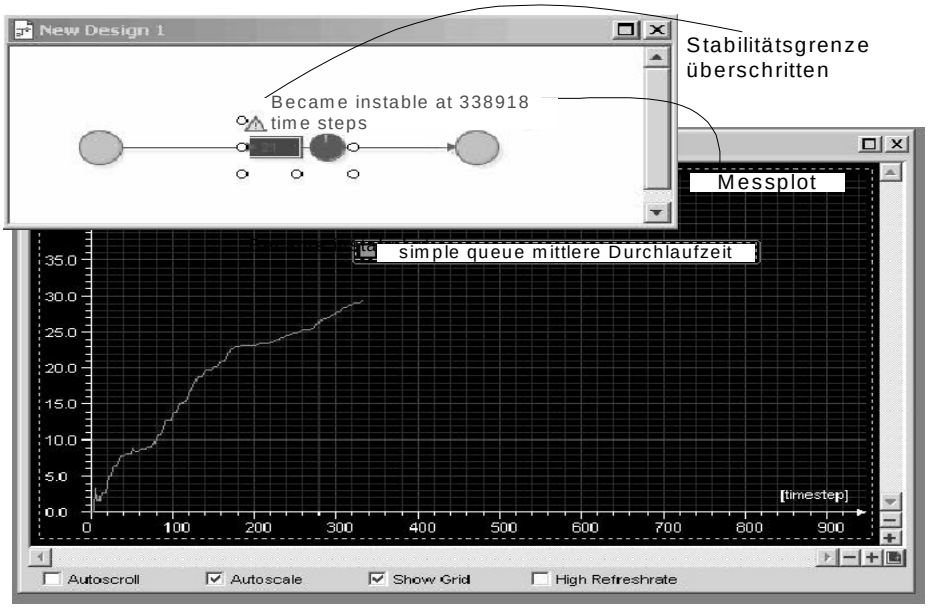

Abbildung 5: Instabile M/M/1 Queue

Zur einfachen Untersuchung bestimmter Systemeigenschaften können durch die Studierenden über den Eigenschaftseditor Systemparameter interaktiv geändert und deren Auswirkungen im weiteren Verlauf des Messplots beobachtet werden.

Neben der *SimpleQueue*, *Source*, *Sink*, *Splitter* und *Merger* stehen den Studierenden weitere Komponenten, wie z.B. *Multiserver*, *Multiqueue*, *ThinkBuffer* sowie Methoden zur Messwerterfassung zur Verfügung. So lassen sich auch komplexere Systeme mühelos modellieren.

4 Zusammenfassung und Ausblick

Mit dem Werkzeug *QueueSim* ist eine Simulationsumgebung entstanden, die nicht nur in der Lehre zur Untersuchung komplexer Wartesysteme eingesetzt werden kann. Durch die interaktive Gestaltung von Modellierung und Simulation lassen sich Modelle quasi spielerisch aufbauen und untersuchen. Aufbauend auf eine Einführungsveranstaltung in die Leistungsbewertung von Rechnersystemen ohne eine allzu grosse Vertiefung in die mathematischen Grundlagen lassen sich dennoch die Eigenschaften von Wartesystemen untersuchen und auf praktische Problemstellungen anwenden. Erste Erfahrungen im Umgang mit der Simulationsumgebung waren sehr gut. Die Studierenden schätzen den experimentellen Ansatz der Simulation und nehmen auch den Grundlagenstoff besser auf.

Durch die bisherigen positiven Erfahrungen bestärkt planen wir, einen Simulationsbaukasten zur Untersuchung von Computernetzen und ihren Protokollen zu erstellen.

Literaturverzeichnis

[Ch00] Chowdhury, Dhiman Deb: High Speed LAN Technology. Springer-Verlag, 2000.

[FL79] Frank, Martin; Lorenz, Peter: Simulation diskreter Prozesse. VEB Fachbuchverlag Leipzig, 1979.

[Gu99] Gunther, Neil J.: The Practical Performance Analyst. MacGraw-Hill, 1999.

[Kl75] Kleinrock, Leonhard: Volume1: Theory. John Wiley, 1975.

[MKR02] Maucher, Johannes; Kunz, Gunnar; Rinkel, Andreas: UMTS EASYCOPE: A tool for UMTS Network and Algorithm Evaluation. International Zürich Seminar on Brodband Communications, 2002.

[MS03] Moser, Christian; Schelldorfer, Martin: Entwicklung eines GUI zur Darstellung und Animation eines Warteschlangensystems. Studienarbeit der Abteilung Informatik der HSR, Schweiz, 2003.

[ZT03] Zwicker, Raphael; Trindler, Jonas: Entwicklung eines Kernsimulators zur Simulation von Wartesystemen. Studienarbeit der Abteilung Informatik der HSR, Schweiz, 2003.