

Werner-Heisenberg-Gymnasium Garching

Kollegstufe 2005/07
Facharbeit aus dem Leistungskurs Mathematik

Diskrete Simulationen in Theorie und Anwendung

Kursleiter: Herr Lortz
Kollegiat: Julius Adorf

Abgegeben am: 26. Januar 2007

Inhaltsverzeichnis

1	Computersimulationen	1
1.1	Einleitung	1
1.2	Begriffsbildung	1
1.3	Statistik und Beobachtung	3
1.4	Ziel einer Simulation	4
1.5	Grenzen der Simulation	4
2	Diskrete Simulation	5
2.1	Varianten	5
2.2	Komponenten	6
3	JDSIM - ein Simulationsprogramm	8
3.1	Eignung der Programmiersprache Java	8
3.2	Eigenschaften	8
4	Eine Flughafensimulation	9
4.1	Fragestellung	9
4.2	Das System "Flughafen"	9
4.3	Modellerstellung	10
4.4	Auswertung	11
5	Ausblick	14
A	Anhang	16
A.1	Inhalt der beigelegten CD-ROM	16
B	Selbstständigkeitserklärung	17

1 Computersimulationen

1.1 Einleitung

“Simulationen eröffnen einen Zugang zu Bereichen, die für herkömmliche Experimente zu klein oder zu groß, zu schnell oder zu langsam, zu gefährlich oder zu teuer sind.” [GG, S. 3]. Deshalb finden Simulationen Anwendungen in der Wissenschaft (Astronomie), im Lehrbereich (Pilotentraining), im Maschinenbau (Motorentechnik, Ingenieurwesen), beim Militär oder in der Wirtschaft (Prozessoptimierung, Logistik).

Bei einer Simulation wird ein reales System auf ein Modell abgebildet, mit dem man experimentieren kann. Die Ergebnisse lassen Rückschlüsse auf das zugrundeliegende System zu.

Simulationen lassen sich mit verschiedenen Mitteln durchführen. Der Computer bietet sich jedoch an: Mit ihm lassen sich große Datenmengen effizient verarbeiten und die erstellten Modelle können leicht geändert werden.

1.2 Begriffsbildung

Bei Simulationen verwendet man einige Schlüsselbegriffe: Systeme, Modelle und Zustände. Auf diesen Begriffen baut die Theorie auf.

1.2.1 Systeme

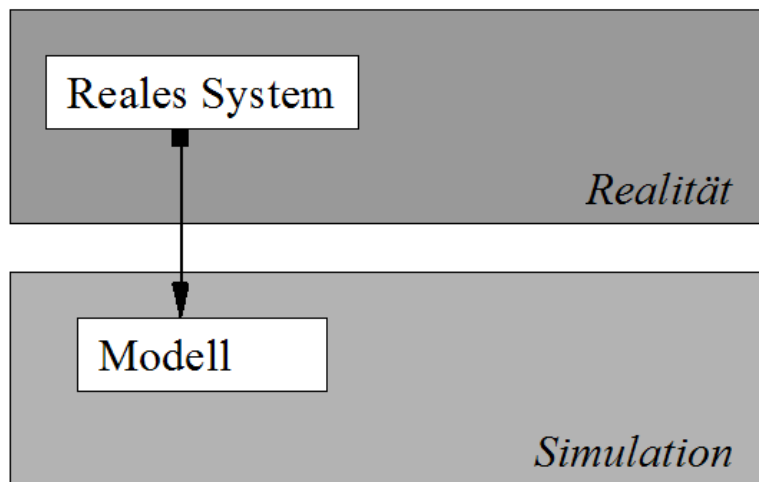
Ein System ist ein Ausschnitt aus der Realität. Es kann mit folgenden Begriffen beschrieben werden:

- *Objekte*: Objekten sind Eigenschaften und bestimmte Verhaltensweisen zugeordnet. Oft werden Objekte in Unterobjekte gegliedert.¹
- *Systemzustand*: Zu jedem Zeitpunkt hat das System einen bestimmten Zustand. Wäre das zugehörige Modell rein deterministisch, so könnten anhand des aktuellen Zustands alle zukünftigen Zustände des Systems vorhergesagt werden.

¹Diese Art der Zusammensetzung von Objekten nennt man *Komposition*.

- *Systemgrenzen*: Diese Grenzen definieren, wo der betrachtete Ausschnitt aus der Realität aufhört.
- *Offenes/Geschlossenes System*: Je nachdem ob das System von außen beeinflusst werden kann oder nicht, spricht man von einem offenen oder geschlossenen System.

1.2.2 Modelle



Ein System wird mit einer bestimmten Methode auf ein Modell abgebildet.

Ein Modell bildet ein reales System ab. Es stellt dabei stets eine idealisierte Vereinfachung des Originals dar [PK05, S. 6]. Innerhalb eines Modells kann man das zu lösende Problem oft in viele kleine überschaubare Teilprobleme zerlegen, indem man das Verhalten einzelner Objekte - welches meistens hinreichend bekannt ist - beschreibt. Die Simulation zeigt dann, wie sich dieses Modell mit fortlaufender Zeit verhält.

Es gibt nicht ein einziges, optimales Modell für ein bestimmtes System; vielmehr wählt man ein Modell so aus, dass sich mit ihm die Fragestellung beantworten lässt. Die Wahl eines geeigneten Modells ist von entscheidender Bedeutung für den Erfolg einer Simulation - also dem Erhalt von aufschlussreichen Ergebnissen.

1.3 Statistik und Beobachtung

1.3.1 Eingabedaten

Für ein Modell werden verschiedene Daten benötigt, die den Grundzustand und das Verhalten der einzelnen Objekte beschreiben. Wenn man nun beispielsweise ein Dartspiel simulieren möchte, stellt man fest, dass die geübteren Spieler besser treffen. Das heißt aber noch nicht, dass die besseren Spieler *immer* genauer werfen als die ungeübten - ein Treffer ist lediglich wahrscheinlicher.

In einem Modell müssen Abläufe, wo der Zufall eine Rolle spielt, stochastisch beschrieben werden. Folglich muss eine stochastische Verteilungsfunktion gefunden werden, die das Modellverhalten beschreibt. Um herauszufinden, welche Verteilungsfunktion das Verhalten möglichst treffend beschreibt, betreibt man häufig Statistik im realen System und überträgt die Ergebnisse auf das Modell.

1.3.2 Datenerfassung

Es sollen diejenigen Daten gewonnen werden, mit denen sich die Fragestellung beantworten lässt. Dazu ist es normalerweise notwendig eine bestimmte Größe oder sogar den Zustand des gesamten Modells an mehreren Zeitpunkten festzuhalten. Die entstandene Zeitreihe gibt Aufschluss über die *zeitliche Entwicklung* des Modells. Aus den Zeitreihen können aber auch für eine zufällige Variable Kenngrößen berechnet werden, wie z.B. den Mittelwert, die Varianz, das Minimum oder das Maximum.

Sogenannte Logs können chronologisch Nachrichten speichern, die den Simulationsablauf beschreiben.

Da sich der Systemzustand einer Simulation normalerweise erst nach einiger Zeit stabilisiert, ist es gelegentlich sinnvoll mit der Aufzeichnung der Daten erst nach einer "Aufwärmphase" zu beginnen [PK05, S. 174].

1.3.3 Datenauswertung

Die Fragestellung lässt sich (hoffentlich) durch die *Interpretation* der gewonnenen Daten beantworten. Dieser Prozess kann automatisiert werden,

jedoch ist es häufig so, dass die Interpretation dem Menschen überlassen ist - für ihn müssen große Datenmengen erst einmal reduziert werden.

1.3.4 Visualisierung

Die grafische Darstellung eines Simulationsablaufs hilft das Modell zu überprüfen. Insbesondere macht sie Zusammenhänge leichter erkennbar. Eine grafisch dargestellte Zeitreihe ist wesentlich aussagekräftiger als eine endlose Zahlenkolonne.

1.4 Ziel einer Simulation

Ziel einer Simulation ist es immer, mit den gewonnenen Ergebnissen Rückschlüsse auf vom Modell auf das reale System zu ziehen. Der Nutzen, d.h. die Verlässlichkeit der Daten hängt zum einen von der Qualität des gewählten Modells ab, zum anderen von den statistischen Unsicherheiten².

1.5 Grenzen der Simulation

Es darf nicht unerwähnt bleiben, dass einer Computersimulation auch Grenzen gesetzt sind. Ein paar der auftauchenden Probleme seien hier aufgelistet.

Mängel am Modell Unzulässige Vereinfachungen führen zu falschen Ergebnissen und machen diese somit wertlos.

Intransparenz Das Modell ist nicht einfach durch eine dritte Person zu überprüfen.

Fehleranfälligkeit Ein kompliziertes Modell begünstigt Fehler bei der Modellerstellung.

Siehe "The Java Simulation Handbook" [PK05, S. 20] für eine ausführlichere Liste.

²Man spricht hierbei von systematischen und zufälligen Fehlern.

2 Diskrete Simulation

Der Begriff “diskret” steht im Gegensatz zu “kontinuierlich” und ist im mathematischen Sinne zu verstehen. Fast alle Vorgänge in der uns bekannten Welt sind kontinuierlich: Zwischen zwei Orten gibt es stets einen Ort, der dazwischenliegt und zeitliche Übergänge verlaufen fließend (es gibt kein kleinstes Zeitintervall Δt).

Eine kontinuierliche Simulation, die die Zeit in beliebig kleine Δt aufteilt, ist nur auf analogen Computern möglich, der aus analogen Schaltkreisen besteht [UVA]. Der digitale Rechner hat sie jedoch verdrängt.

Die diskrete Simulation teilt die Zeit in Intervalle ein. Der Modellzustand ändert sich schrittweise. Zwischen zwei Zeitpunkten ändert sich der Modellzustand hingegen nicht. Durch die zeitliche Abfolge aller Zustandsänderungen lässt sich der Ablauf einer diskreten Simulation vollständig beschreiben.

2.1 Varianten

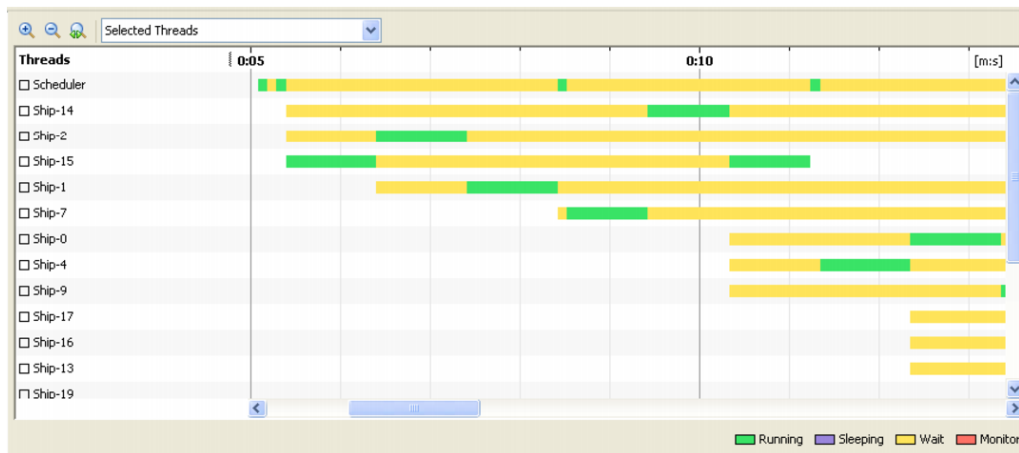
Es gibt verschiedene Varianten der diskreten Simulation. Am häufigsten werden ereignisorientierte und prozessorientierte Simulationen implementiert [PK05, S. 98].

2.1.1 Ereignisgesteuerte Simulation

Der gesamte Ablauf einer ereignisgesteuerten Simulation wird durch die Abfolge von Ereignissen beschrieben, die logisch miteinander verknüpft sind. Die Simulation eines Reflexes ließe sich z.B. als die Abfolge der Ereignisse “wahrnehmen”, “verarbeiten”, “reagieren” darstellen.

2.1.2 Prozessorientierte Simulation

In der prozessorientierten Simulation laufen mehrere Prozesse quasi-parallel ab. Tatsächlich sind die einzelnen Prozesse so synchronisiert, dass immer genau ein Prozess aktiv ist - nur dieser darf den Modellzustand verändern. Die folgende Abbildung illustriert dies.



Übersicht im NetBeans 5.5 Profiler über die laufenden Prozesse während einer Simulation: Es ist immer genau ein Prozess aktiv.

2.1.3 Kombination aus beiden

Eine zu einseitige Sichtweise bringt oftmals Nachteile mit sich. Beispielsweise tut man sich bei einer rein prozessorientierten Fabriksimulation schwer, einen Stromausfall oder einen Feueralarm zu simulieren. Der Stromausfall müsste dann als eigener Prozess modelliert werden, was recht konstruiert und unnatürlich wirkt.

Deshalb kombiniert man die beiden Verfahren, um die Vorteile beider Methoden zu erhalten. Mit einer Kombination aus der ereignis- und der prozessorientierten Simulation kann man das Fallbeispiel elegant lösen. Der Stromausfall wird zu einem Ereignis, das alle Maschinen der Fabrik (Prozesse) lahmlegt [PK05, S. 134].

2.2 Komponenten

Es gibt einige Komponenten, die in so gut wie jeder Simulation auftreten und somit eine gemeinsame Basis bilden.

2.2.1 Ablaufsteuerung

Die Ablaufsteuerung stellt die Simulationsuhr da. Sie steuert das Fortschreiten der Modellzeit und sorgt dafür, dass die Ereignisse zu den ihnen zugewiesenen Zeitpunkten in der richtigen Reihenfolge eintreten.

Die Zeit läuft dabei nicht kontinuierlich weiter, sie "springt" immer zum nächsten Zeitpunkt, an dem ein Ereignis eintritt. Da nur notwendige Schritte durchgeführt werden, steigert dieser Ansatz die Effizienz.

In der Programmierung nennt man die Ablaufsteuerung üblicherweise "Scheduler".

2.2.2 Objekte

Ein Modell besteht aus mehreren Objekten die miteinander kommunizieren, Ereignisse auslösen oder neue Objekte erschaffen können. Ein einzelnes Objekt wird meistens als "Actor" oder als "Entity" bezeichnet.

3 JDSIM - ein Simulationsprogramm

JDSIM steht für “Java Discrete Simulation Framework”. Die Software bildet die Grundlage für die Erstellung von diskreten Modellen.

3.1 Eignung der Programmiersprache Java

Da ein Modell normalerweise aus mehreren interagierenden Objekten besteht, eignen sich für die Simulation objektorientierte Programmiersprachen besonders. Eine davon ist Java.

“Höhere Programmiersprachen” (wie Java) bringen allerdings den Nachteil mit sich, dass sie in manchen Bereichen nicht so schnell sind wie Programmiersprachen, die sich näher am Maschinencode orientieren.

3.2 Eigenschaften

Das JDSIM-Simulationsprogramm besitzt folgende Eigenschaften:

Robustheit: Fehler werden so früh wie möglich erkannt und so genau wie möglich beschrieben. Logische Fehler im Modell werden normalerweise erkannt.

Flexibilität: JDSIM bietet eine Grundlage für verschiedene Simulationen.

Benutzerfreundlichkeit: Steile Lernkurve, grafische Darstellung, verständlicher Code und ausreichende Dokumentation erleichtern die Arbeit.

4 Eine Flughafensimulation

Die Theorie soll nun in die Praxis umgesetzt werden. Als Anschauungsbeispiel dient ein Flughafenmodell, welches in Java geschrieben ist und das JDSIM-Simulationsprogramm verwendet. Auf die genaue Implementierung soll nicht eingegangen werden.

4.1 Fragestellung

Mit der Flughafensimulation sollen folgende (teilweise verwandte) Fragestellungen untersucht werden:

1. Wie hängt die mittlere Wartezeit eines Flugzeugs mit der Auslastung der Landebahnen und Flugsteige zusammen?
2. Wie stark können die vorhandenen Kapazitäten ausgelastet werden?
3. Welche Ressourcen sind überlastet?
4. Welche Maßnahmen können ergriffen werden, um die Kapazitäten besser auszulasten?
5. Wie können die Wartezeiten verkürzt werden?

Das Modell wird auf diese Fragestellung zugeschnitten. Doch bevor das Modell erstellt werden kann, muss das System "Flughafen" untersucht werden.

4.2 Das System "Flughafen"

4.2.1 Flugzeuge

Flugzeuge können in das System "Flughafen" eintreten. Ein Flughafen stellt folglich ein offenes System dar. Nach der Landung begeben sie sich an einen Flugsteig, wo sie abgefertigt werden bevor sie den Flughafen wieder verlassen. Gelegentlich müssen sie dabei auch warten.

4.2.2 Flughafeneinrichtungen

Ein Flughafen besteht aus Start- und Landebahnen, Rollbahnen, Terminals mit einer Anzahl an Flugsteigen (die oftmals flugzeugtypspezifisch sind) und einem Kontrollteam im Tower, das die Start- und Landegenehmigungen verteilt und die Rollbahnen überwacht. Es gibt üblicherweise Wartungshallen und ein Frachtzentrum.

4.3 Modellerstellung

Das gerade eben beschriebene System wird nun nachmodelliert. Dabei erhebt dieses Beispielmmodell keinen Anspruch darauf, vollständig zu sein (dafür braucht es Expertenwissen aus der Flughafenbranche).

4.3.1 Abstraktionen und Vereinfachungen

Auch beim Flughafenmodell kann man bestimmte Aspekte getrost beiseite lassen und davon abstrahieren. Was man weglässt, hängt allerdings immer von der Fragestellung ab.

Das Modell wird auf Flugzeuge, Lande-/Startbahnen und Flugsteige reduziert.

Außerdem werden die ganzen Möglichkeiten des "air traffic flow management" (absichtliche Verzögerungen, geschickte Warteschleifen zur Stauvermeidung) außer Acht gelassen. Die Wirbelschleppen werden beachtet. Es wird davon ausgegangen, dass auf dem Flughafen in der untersuchten Zeit keine Störungen auftreten (wie Brände, Nebel, Maschinenausfall, Bombenalarm).

4.3.2 Eingabedaten und Anfangsparameter

Für das Flughafenmodell braucht man Eingabedaten. Das Beispielmmodell benötigt grob folgende Parameter:

- Rate der ankommenden Flugzeuge

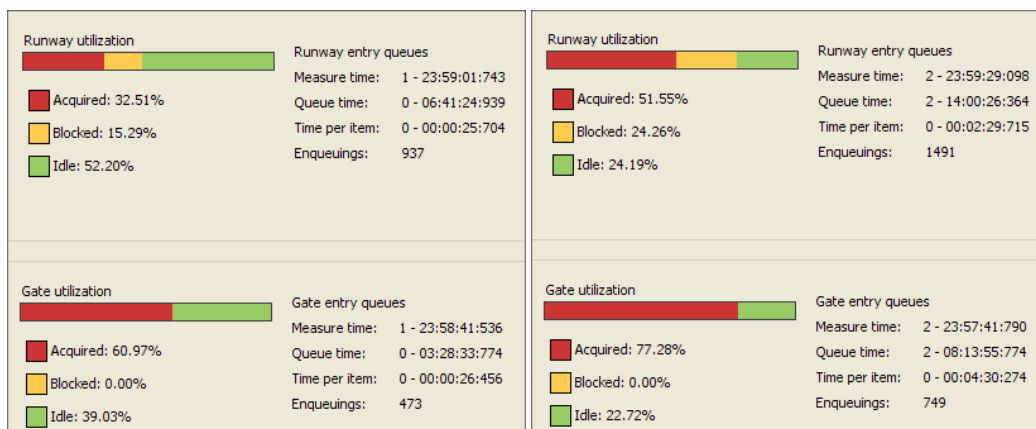
- Verhältnis der Flugzeugtypen
- Anzahl der Landebahnen
- Anzahl der Flugsteige
- Benötigte Zeit pro Start- und Landevorgang
- Verbrachte Zeit auf Rollbahnen
- Benötigte Zeit für die Abfertigung eines Flugzeugs

Dabei genügt es zunächst festzustellen, *welche* Eingaben benötigt werden.³

4.4 Auswertung

4.4.1 Auslastung

Die einzelnen Ressourcen können sehr unterschiedlich ausgelastet sein. Die Ressourcen mit hoher Auslastung sind potenzielle Engpässe⁴, die Wartezeiten verursachen können. Die Auslastung der Start- und Landebahnen sowie die Nutzung der Flugsteige wird über die gesamte Simulation aufgezeichnet. Am Ende der Simulation können die ermittelten Werte grafisch dargestellt werden.



Aufbereitung der Ergebnisse zweier Simulationen. Die Daten werden dabei auf das Wesentliche reduziert.

³Unter <http://www.transtats.bts.gov/> gibt es umfassende Statistiken, die allerdings Ergebnisdaten enthalten. Für unser Flughafenmodell würde eine Analyse den Rahmen sprengen.

⁴Auf Englisch: bottlenecks

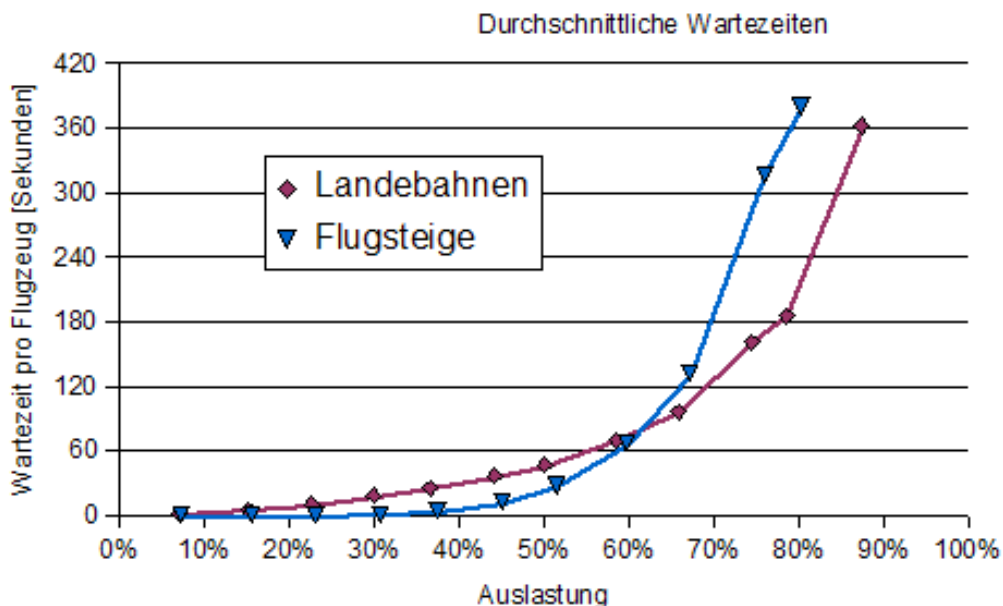
Für eine gründlichere Analyse müssen auch noch die einzelnen Wartezeiten untersucht werden.

4.4.2 Wartezeiten

Nun soll die Frage beantwortet werden, wie sich die Auslastung der einzelnen Stationen auf die mittlere Wartezeit eines Flugzeugs auswirkt. Dabei kann man folgende Zeiten betrachten:

- Wartezeit bis eine Landebahn frei wird
- Wartezeit bis ein Flugsteig frei wird
- Wartezeit bis eine Startbahn frei wird

Die Anzahl der ankommenden Flugzeuge pro Stunde wurde in mehreren Simulationen variiert, um mittlere Wartezeiten zu verschiedenen Auslastungen zu erhalten. Daraus entsteht folgendes Diagramm:



Es ist doch erstaunlich! Die Wartezeiten wachsen keineswegs linear, sondern eher exponentiell.

4.4.3 Fazit

Man kann nun verschiedene Strategien verfolgen, um die Wartezeiten zu verkürzen *und* die Kapazität besser auszunutzen. Diese Ziele stehen nur bedingt im Widerspruch. Es treten nämlich Phänomene auf, die die Wartezeiten selbst bei wenig Auslastung explodieren lassen. Beispielsweise können zufällig mehrere Flugzeuge auf einmal landen wollen und es kommt zu einem Stau. Einige Zeit später kann es passieren, dass alle Landebahnen eine Zeitlang völlig ungenutzt bleiben. Es gilt also, die Varianzen zu minimieren.

Eine effizientere Nutzung der Stationen kann man auf verschiedene Weisen erhalten. Einige seien hier aufgelistet.

- Einführung von Reservekapazitäten.
- Einführung von Warteschleifen und anderen Maßnahmen, um gleichmäßigere Intervalle zwischen zwei Flugzeugen zu erhalten.
- Immer abwechselnd Start und Landung durchführen, damit die Wirbelschleppen nicht so ins Gewicht fallen.

Das Flughafenmodell ermöglicht das Experimentieren mit verschiedenen Anfangsparametern und Strategien. Die Ergebnisse tragen zum Verständnis des echten Systems "Flughafen" bei.

5 Ausblick

Letztendlich ist die Computersimulation immer ein Mittel zum Zweck. Simulationsprogramme sind Werkzeuge, die die Erstellung eines Modells möglichst so vereinfachen, dass sich der Anwender auf die *wesentlichen* Eigenschaften eines Modells konzentrieren kann. Software zu schreiben, die diese Anforderungen erfüllt, bleibt eine Herausforderung.

Meiner Einschätzung nach werden sich also Simulationsprogramme durchsetzen, mit denen man Modelle für verschiedene Anwendungsbereiche grafisch erstellen kann, die den Simulationsverlauf visualisieren und es dem Anwender erlauben, aktiv in den Ablauf einzugreifen. Für rechenaufwändige Simulationen sollten die Simulationsprogramme gridfähig gemacht werden, so dass man sie auf mehrere Rechner verteilen kann.

Die Simulationsprogramme werden von der Weiterentwicklung von Programmiersprachen und Programmiertechniken profitieren. Ich bin gespannt: Was bewegt sich in den nächsten Jahren?

Literatur

- [PK05] Bernd Page, Wolfgang Kreutzer, *The Java Simulation Handbook* (Aachen: Shaker Verlag, 2005).
- [BS87] Jörg Biethahn, Bernd Schmidt, *Simulation als betriebliche Entscheidungshilfe (1)* (Berlin: Springer-Verlag, 1987).
- [GG] *Computersimulationen- Neue Instrumente der Wissensproduktion* (<http://www.sciencepolicystudies.de/dok/expertise-gramelsberger.pdf>, 29 Oct. 2006).
- [LLNRW] Helmut Kohorst, Philipp Portscher, Peter Goldkuhle, *Learn:line NRW - Arbeitsbereich Modellbildung und Simulation* (<http://www.learn-line.nrw.de/angebote/modell/modlist.htm>, 3 Oct. 2006).
- [MUSDS] *Modellbildung und Simulation dynamischer Systeme* (<http://modsim.hupfeld-software.de>, 3 Oct. 2006).
- [UVA] *Analog computers* (<http://www.science.uva.nl/museum/AnalogComputers.html>, 24 Jan. 2007).
- [GADGET] *Cosmological simulations with GADGET* <http://www.mpa-garching.mpg.de/gadget>, 10 Jan. 2007
- [AIRNAV] *AirNav: Airport Information* (<http://www.airnav.com/airports/state>, 17 Nov. 2006).

A Anhang

A.1 Inhalt der beigelegten CD-ROM

- **JDSIM-Framework** - Java Discrete Simulation Framework. Benötigt die Installation von Java 6.
- **AVISIM** - das Beispielprogramm der Flughafensimulation. Benötigt die Installation von Java 6.
- **Java 6** - ermöglicht, Java-Programme auszuführen.
- **NetBeans 5.5** - eine Open-Source-Entwicklungsumgebung (IDE) speziell für Java. Dieses Programm ist für die Ausführung des Beispielprogramms *nicht* nötig.
- **Quellennachweise**

B Selbstständigkeitserklärung

Ich erkläre hiermit, daß ich die Facharbeit ohne fremde Hilfe angefertigt und nur die im Literaturverzeichnis angeführten Quellen und Hilfsmittel benützt habe.

Ort,

Datum

Unterschrift des Schülers