KfK 5310
April 1994

3. Treffen des AK Werkzeuge für Simulation und Modellbildung in Umweltanwendungen

am 28.10. - 29.10.1993 in Kassel/Witzenhausen

H. B. Keller, R. Grützner, J. Benz (Hrsg.)
Institut für Angewandte Informatik
Projekt Schadstoff- und Abfallarme Verfahren

Kernforschungszentrum Karlsruhe

Kernforschungszentrum Karlsruhe

Institut für Angewandte Informatik
Projekt Schadstoff- und Abfallarme Verfahren

KfK 5310

3. Treffen des AK Werkzeuge für Simulation und Modellbildung in Umweltanwendungen

am 28.10. - 29.10.1993
in Kassel / Witzenhausen

Hubert B. Keller, Rolf Grützner*,
Joachim Benz+
(Hrsg.)

* Universität Rostock, Fb Informatik

+ Gesamthochschule Kassel, Fb 11

Kernforschungszentrum Karlsruhe GmbH, Karlsruhe

Kernforschungszentrum Karlsruhe GmbH
Postfach 3640, 76021 Karlsruhe

ISSN 0303-4003

Zusammenfassung:

Dieser Bericht stellt die Beiträge des 3. Treffens des Arbeitskreises "Werkzeuge für Simulation und Modellbildung in Umweltanwendungen" des Fachausschuß 4.6, "Informatik im Umweltschutz", der Gesellschaft für Informatik (GI) vom 28.10. - 29.10.1992 in Kassel / Witzenhausen in schriftlicher Form dar.

3rd Meeting of the WG
Tools for Simulation and Modelling in Environmental Applications

Abstract:

This report contains the papers of the 3rd meeting of the working group " Tools for Simulation and Modelling in Environmental Applications", of the section 4.6, "Computer Science for Environmental Protection", of the Society for Computer Science (GI), which took place at Kassel / Witzenhausen on 28th - 29th of october in 1993.

Vorbemerkungen

Die Modellbildung und Simulation ist eine wichtige Methode zur Behandlung von Problemen im Umweltschutz. Sie gestattet, die Wirkungen der anthropogenen Aktivitäten auf die Umwelt mehr oder weniger exakt vorherzusagen bzw. abzuschätzen. Mit ihrer Hilfe ist es möglich, den Umweltzustand sowie seine Veränderungen zu untersuchen und vorherzusagen. Auch die Einwirkungen von Produktionsprozessen und anderen menschlichen Aktivitäten sowie die Wirksamkeit von Umweltschutzmaßnahmen können mit dieser Problemlösungsmethode erfaßt bzw. abgeschätzt werden. Damit bildet die Modellierung und Simulation ein wichtiges Werkzeug, um u.a. bei der Untersuchung des Umweltzustandes, bei Planungsaufgaben, bei der umweltgerechten Produktionssteuerung und den erforderlichen Umweltverträglichkeitsprüfungen eingesetzt zu werden.
Solche Analysen werden jedoch durch die Komplexität der Systeme und dem häufig noch fehlenden Wissen über diese erheblich erschwert. Daraus folgt, daß Modelle auf der Basis unscharfer Systemkenntnisse und fehlender Daten aufgestellt und genutzt werden müssen. Durch geeignete Methoden und Werkzeuge der Systemtheorie und der Informatik wird versucht, auch diesen Bedingungen Rechnung zu tragen. Methodische, funktionelle, strukturelle und deskriptive Fragen der Modellbildung standen neben Anwendungen im Blickfeld des zweitägigen Treffens des Arbeitskreises 5 "Werkzeuge für Simulation und Modellbildung in Umweltanwendungen" der GI-Fachgruppe 4.6.1 : Informatik im Umweltschutz.
In 8 Vorträgen, verbunden mit sehr ausführlichen Diskussionen zu den Vorträgen und in den Pausen, wurden folgende Themen behandelt: qualitative Systemanalyse, neuronale Netze bei der Prozeßsteuerung, objektorientierte Datenbanken und Systemsimulation, Dokumentation von Ökosystemmodellen, Simulationsumgebungen und Verkehrssimulation.
Das Treffen fand an der Gesamthochschule Kassel in Witzenhausen bei Kassel statt. Herr Dr. J. Benz hat, als örtlicher Organisator, für die erfolgreiche Durchführung in der wunderschönen Umgebung gesorgt. Dafür sei ihm ganz herzlich gedankt.
Gedankt sei auch dem Kernforschungszentrum Karlsruhe und Herrn Dr. H. B. Keller für die Bereitschaft, die Beiträge des Treffens wieder in den KfK-Berichten zu veröffentlichen sowie Herrn Dr. J. Benz für das Zusammentragen der Beiträge.
Es ist vorgesehen, das nächste Treffen im Juni 1994 durchzuführen.

Prof. Dr. habil. R. Grützner
Leiter Arbeitskreis 5

Inhaltsverzeichnis

Unterstützung der Technikfolgenforschung durch Software-Werkzeuge

Dr. Michael H. Ruge
Siemens AG
Zentralabteilung Forschung und Entwicklung
Technikfolgenforschung

Ausgangslage:

Im Mittelpunkt einer Firma - wie auch der Siemens AG - stehen Produkte und Technologien. Durch neue Auswirkungsgebiete dieser Technologien oder Produkte auf die Umwelt, Gesellschaft und Politik sind neben traditionellen Unternehmenszielen Werte hinzugekommen, denen unternehmerisches Handeln zunehmend gerecht werden muß. Dabei ist Technikfolgenabschätzung ein Instrument, um auf diese neue Situation zu reagieren.

Technology Assessment oder Technikfolgenabschätzung ist ein Begriff, der 1966 zum ersten Male in einem Bericht über die sekundären Auswirkungen technischer Innovationen des amerikanischen House of Representatives benutzt wurde /3/. Dies führte 1972 zur Gründung des Office of Technology Assessment (OTA) in Washington D.C. Dieses definiert Technology Assessment wie folgt:

Technology assessment is a term used to identify a process for generating accurate, comprehensive, and objective information about technology to facilitate its effective social management by political decision-makers.

Die Aufgabe des OTA besteht mithin darin, technische Inhalte dem amerikanischen Kongreß verständlich zu machen, der über einen geringen ingenieurwissenschaftlichen Hintergrund verfügt. Trotz gewisser Anfangsprobleme handelt es sich bei dem OTA mittlerweile um eine anerkannte Institution, dessen Meinung auch häufig von der Industrie nachgefragt wird. Auch war das OTA maßgebend bei der Errichtung des Büros für Technikfolgenabschätzung und -bewertung beim Deutschen Bundestag beteiligt.

In Analogie zum OTA versteht sich das Referat für Technikfolgenforschung bei Siemens als eine Institution, die den Vernetzungscharakter zwischen technischen und nicht technischen Gebieten in Form von Projekten mit Partnern in Bereichen der Siemens AG und an Instituten/Hochschulen untersucht. Dazu gilt es, sozio-technische Systeme präzise zu modellieren und Aussagen über ihre Überlebensfähigkeit /5/ zu gewinnen. Diese sogenannte Sensitivitätsanalyse /6/ wurde u.a. in einer Studie für die Siemens Solar GmbH angewandt.

Zur Zeit geschehen solche Analysen weitgehend heuristisch, indem etwa Aussagen wie: "Steigt der Energieverbrauch, so steigt auch die Umweltbelastung" verbal von Experten formuliert und beurteilt werden. Eine Simulation des Systems findet nicht statt. Daneben gibt es aber etablierte quantitative Verfahren, die sich beim Modellieren und Simulieren rein, oder zumindest überwiegend technischer

Systeme bewährt haben. In diesem Zusammenhang sei System Dynamics erwähnt, eine Methode zur Simulation gekoppelter zeitabhängiger Differentialgleichungssysteme.

Simulationsarbeiten:

In der Gruppe für Technikfolgenforschung wird an der Unterstützung der Modellierung sozio-technischer Systeme durch Software gearbeitet. Bei dem Prototypen CATS (Computer-Aided Technology Assessment Software) handelt es sich um eine offene Umgebung, die in der objektorientierten Programmiersprache SMALLTALK entwickelt wird und in die daher jederzeit neue Simulationsalgorithmen integriert werden können. Bei der Modellierung und Simulation dynamischer Systeme unterstützt ein graphischer Editor die Eingabe der Netzwerke, die aus Variablen und deren Wechselbeziehungen bestehen und die in interdisziplinären Workshops ermittelt werden.

Im Falle einer quantitativen Simulation wird das Modell durch Definition der Werte der Variablen und die Spezifikation der Änderungsraten der Wechselbeziehungen für die Simulation vorbereitet. Während der Animation berechnen sich in jedem Simulationsschritt die Änderung der Variablen aus der Summe der Änderungsraten der auf sie wirkenden Funktionen.

Die Ähnlichkeit der quantitativen Simulation in CATS mit System Dynamics bedeutet, daß auch dessen Vor- und Nachteile geteilt werden. Ein wesentliches Problem liegt in der Unschärfe bei der Quantifizierung der zugrundeliegenden Daten, besonders dann, wenn es neben technischen Größen auch soziale Komponenten zu berücksichtigen gilt.

Um diese Methoden zu kombinieren, d.h. auch für qualitative Systeme durch präzise Modellierung systemische Aussagen zu gewinnen, entwickelt der Autor einen qualitativen Kausal-Analyse-Algorithmus, der es ermöglicht, sozio-technische Systeme zu analysieren und später auch zu simulieren. Zur Zeit beschränkt sich der Output dieser Simulationssoftware auf die Angabe einer sogenannten initiellen Antwort, die das sofortige Verhalten des Systems auf eine Störung widerspiegelt, und einer sogenannten Ultimate response, die das Verhalten des Systems nach dem Einschwingen qualitativ beschreibt.

CATS unterstützt Qualitative Analysis of Causal Feedback (QUAF), das erfolgreich zur Analyse von kleinen Störungen in chemischen und verfahrenstechnischen Prozessen eingesetzt wird /2/. Schwerpunkt ist eine Betrachtung der Rückkopplungskreise des Systemmodells. Diese ermöglicht ein erstes Systemverständnis und hilft so, Fehler beim Aufbau von Modellen zu vermeiden. Der Algorithmus basiert auf Arbeiten des Massachussetts Institute of Technology in den achtziger Jahren und wurde erfolgreich in der chemischen Verfahrenstechnik zur Ermittlung von Prozeßstörungen angewandt.

Einschränkend muß aber gesagt werden, daß nur das Verhalten des Systems bezüglich der Einwirkung einer Störung am M.I.T. untersucht wurde. Der Autor hat in seiner wissenschaftlichen Publikation für die European Simulation MultiConference ESM 93 gezeigt, daß prinzipiell nichts gegen eine Verallgemeinerung auf mehrere Störgrößen spricht /4/. Schließlich ist auch zu bedenken, daß in sozio-technischen Systemen durchaus mehrere Störungen eintreten können. Dies unterscheidet diese zu den speziellen in der Verfahrenstechnik betrachteten Systemen, wo es primär darum geht, in Sekundenschnelle die Ursache z.B. für einen sich unerwartet ändernden Meßwert zu ermitteln, die in einer geborstenen oder verstopften Leitung liegen könnte, um den entstehenden Schaden zu begrenzen.

Zusammenfassung und Ausblick:

Technikfolgenforschung ist eine Dienstleistung zur Entscheidungsunterstützung. Dies beinhaltet die

- Formulierung der Problemstellung in interdisziplinären Workshops
- Modellierung von Entscheidungssituationen
- Analyse und Simulation des Systemmodells auf die zu untersuchende(n) Fragestellung(en)
- Ableitung von Maßnahmen zur Erreichung der gewünschten Ziele
- Darstellung der Konsequenzen von einzuleitenden Maßnahmen, sowie
- Begründung aller im Modellierungs- und Entscheidungsprozeß unternommenen Schritte.

Es ist ersichtlich geworden, daß zur Erreichung dieser Ziele - unter denen im besonderen die Modellierung, Analyse und Simulation sozio-technischer Systeme zu nennen ist - eine auf solche Systeme zugeschnittene Software benötigt wird. CATS oder Computer-Aided Technology Assessment Software ist ein Prototypen-Simulations-System, das als Schritt in diese Richtung zu werten ist und vom Autor mit Schwerpunkt auf qualitative Simulation kontinuierlich weiterentwickelt wird.

CATS basiert primär auf den Arbeiten des M.I.T., enthält aber zusätzlich dem System Dynamics verwandte Komponenten. Da die qualitative Simulation in CATS auf signierten gerichteten Graphen aufbaut und da es bereits Ansätze gibt, wie sich diese mit System-Dynamics-Modellen kombinieren lassen /1/, könnte ein Ziel einer weiteren Softwareentwicklung sein, klassische quantitative Ansätze wie System Dynamics mit neueren qualitativen Methoden zu verbinden.

Literatur:

1. Dolado, J. J., Qualitative Simulation and System Dynamics. In: *System Dynamics Review,* 1992. 8(1): pp. 55-81.

2. Oyeleye, O. O. and Kramer, M. A., *Qualitative Simulation of Chemical Process Systems: Steady State Analysis.* AIChE Journal, 1988. 34(9): pp. 1441-1454.

3. Paschen, H., Gresser, K., and Conrad, F., Stand des Technology Assessment in verschiedenen Ländern und auf internationaler Ebene. In: *Technology Assessment - Technologiefolgenabschätzung.* 1978, Campus Verlag: Frankfurt. pp. 81-96.

4. Ruge, M. H. and Siepmann, E., Technology Assessment and Qualitative Simulation. In: *Modelling and Simulation ESM 93.* 1993, Society for Computer Simulation International: Lyon, Frankreich. pp. 139-144.

5. Vester, F., *Das kybernetische System Leben.* Bild der Wissenschaft, 1977. 3-1977: pp. 86-96.

6. Vester, F. and v. Hesler, A., *Sensitivitätsmodell (in English and German).* Forschungsbericht 80-101 040 34. 1980, Umweltforschungsplan des Bundesministers des Innern, Regionale Planungsgemeinschaft, Untermain, Frankfurt am Main.

SIEMENS

Technology Assessment

(in short: TA)

. . . is a term used to identify a process for generating accurate, comprehensive, and objective information about technology to facilitate its effective social management by political decision-makers.

OTA

. . . is the well-planned, systematic, organized procedure, which

- *analyzes a technology*
- *forecasts the direct and indirect technical, economical, ecological, human, social, health and other consequences of this technology and its possible alternatives*
- *assesses these consequences by means of defined aims and values and projects further desirable developments*
- *deduces and elaborates on course of action and organization*

in order to assist in making reasonable and realizable decisions.

VDI

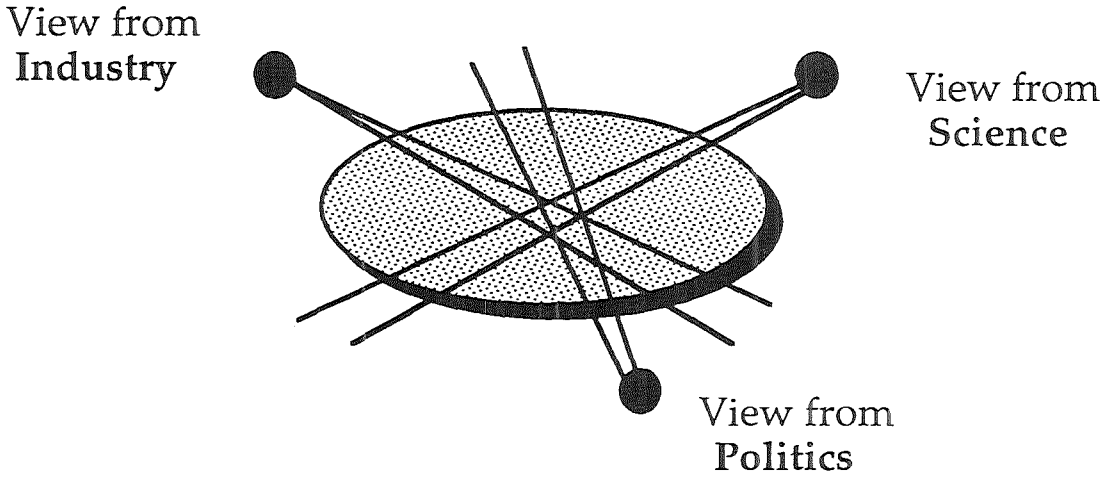
Why Technology Assessment?

In general new technologies are ...

- **More**
  - *expensive*
  - *ethically difficult*
  - *politically sensitive*
  - *complex*
  - *connected*
  - *momentum*
- **Less**
  - *predictable*

SIEMENS

Different Views of Reality



- *Install interdisciplinary teams*
- *Develop a hollistic model of the cross-linked domain of common interest*
- *Search for accord; do not strive for absolute correctness*

Methods Currently Available

Scenario Technique

Sensitivity Analysis

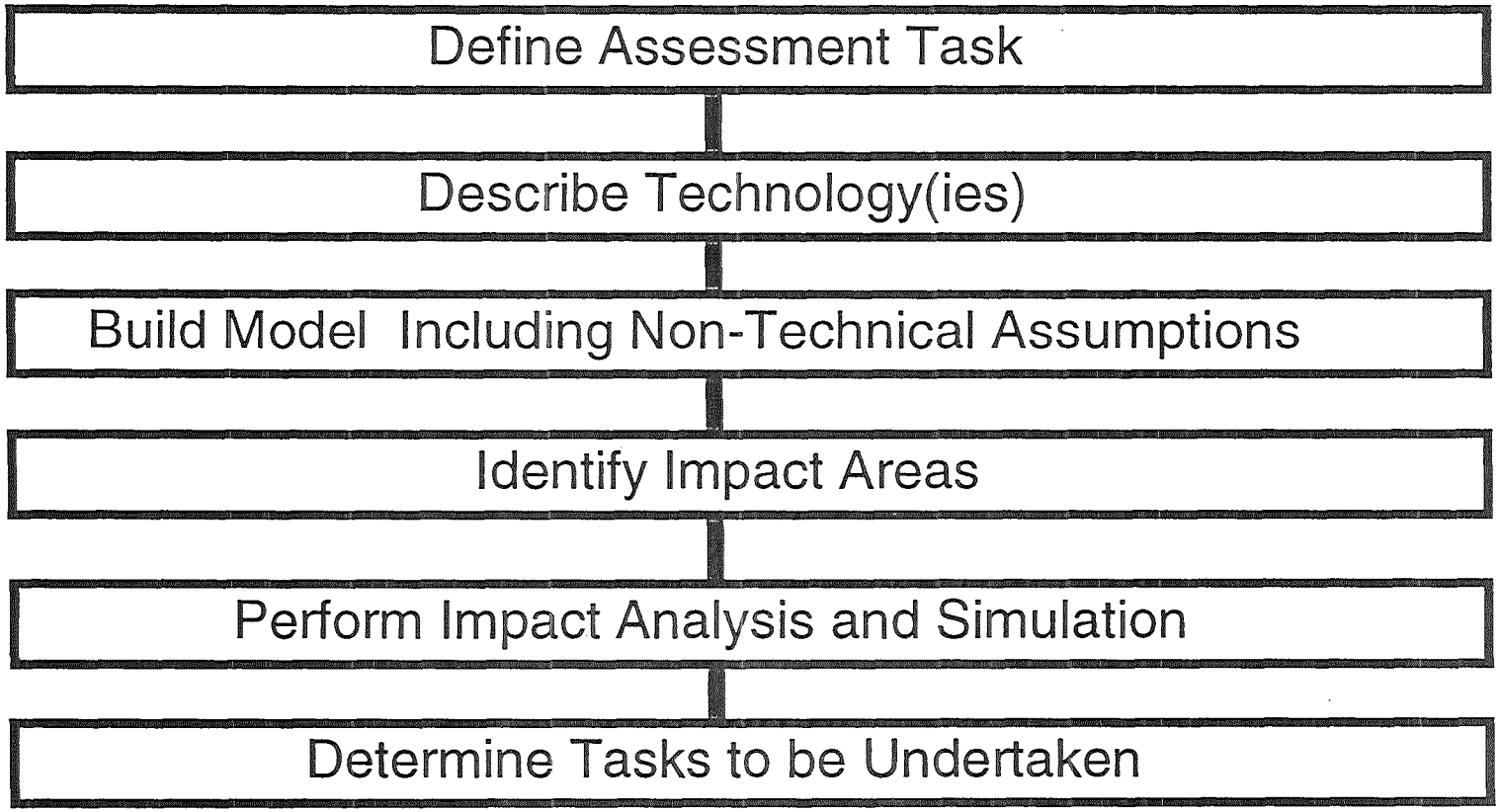
Technology Comparisations

Qualitative Analysis of Causal Feedback

System Dynamics

SIEMENS

Major Steps in a TA Project

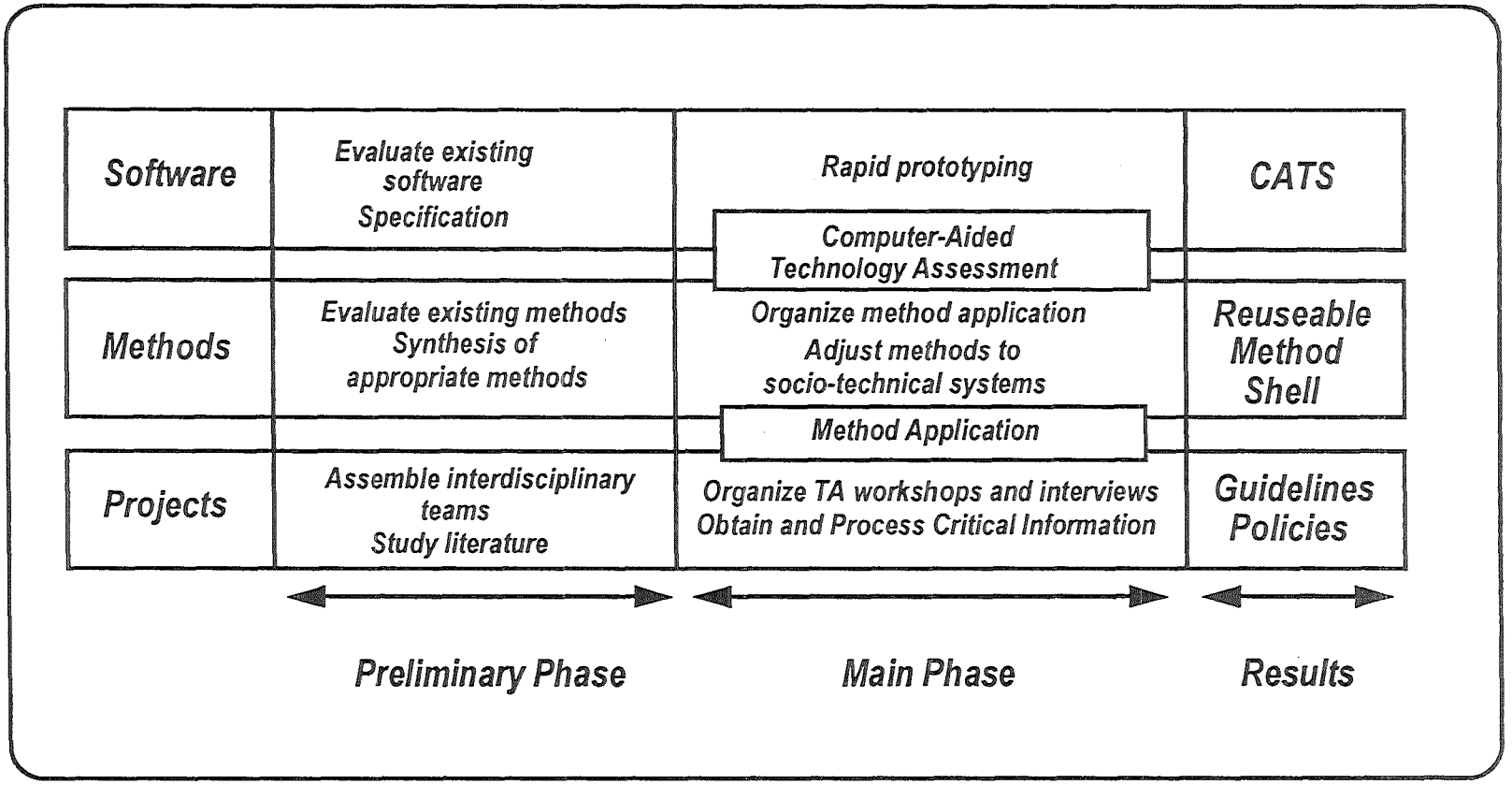


Research Tasks
and
Future Activities

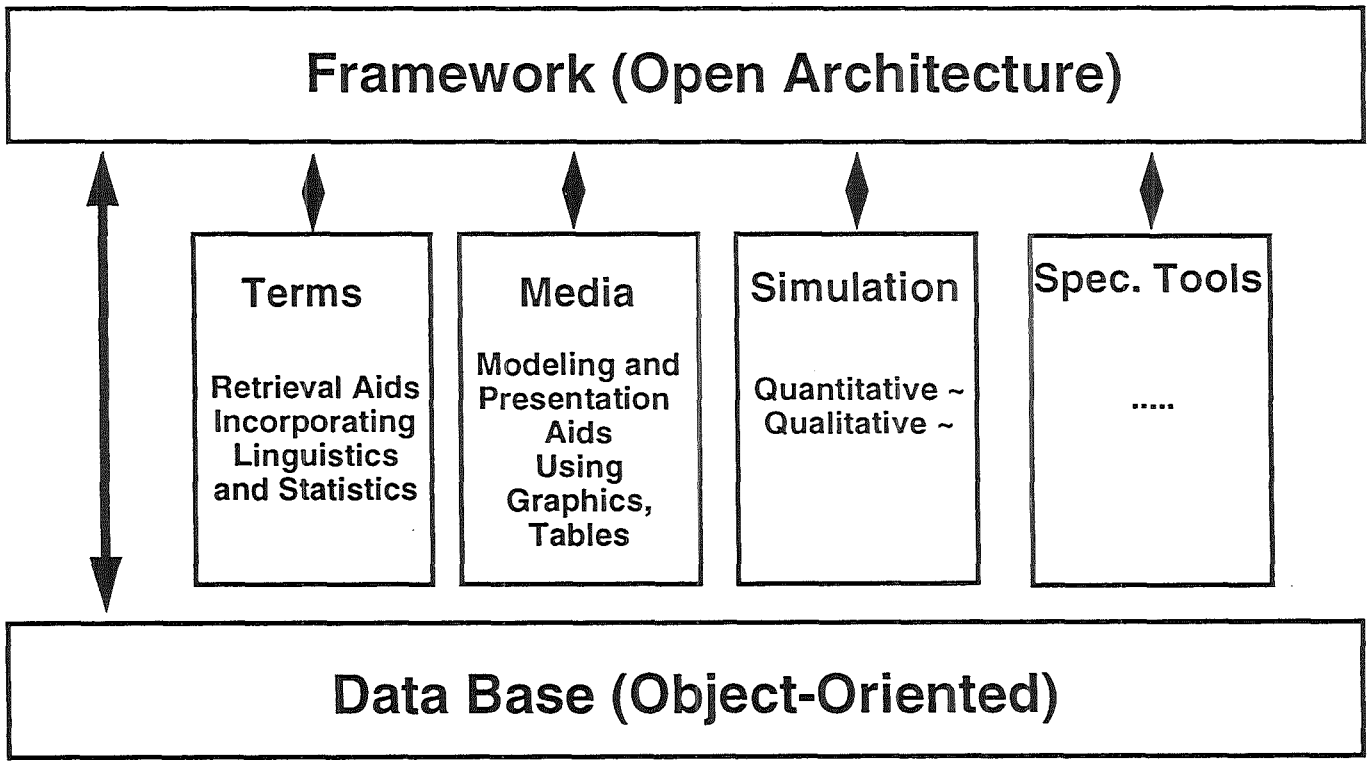
- *Develop Sound Products by Applying Technology Assessment Early on*
- *Provide Technology Assessment Services (Projects, Training, Assistance)*
- *Develop Software Assisting TA Process (CATS)*
- *Contribute to Today´s TA Research (Communication, Publications)*

SIEMENS

Current Aspects



Computer-Aided Technology Assessment Software CATS

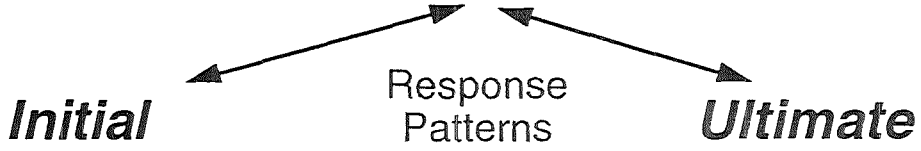


SIEMENS

Qualitative Simulation in CATS

Qualitative Analysis of Causal Feedback (M.I.T.)

- ***Start off with (Qualitative) System of Differential Equations***
- ***Incorporate Algebraic Equations by Reversing to Causality***
- ***Build Signed Directed Graph SDG***
- ***Perform Qualitative Analysis or Simulation***

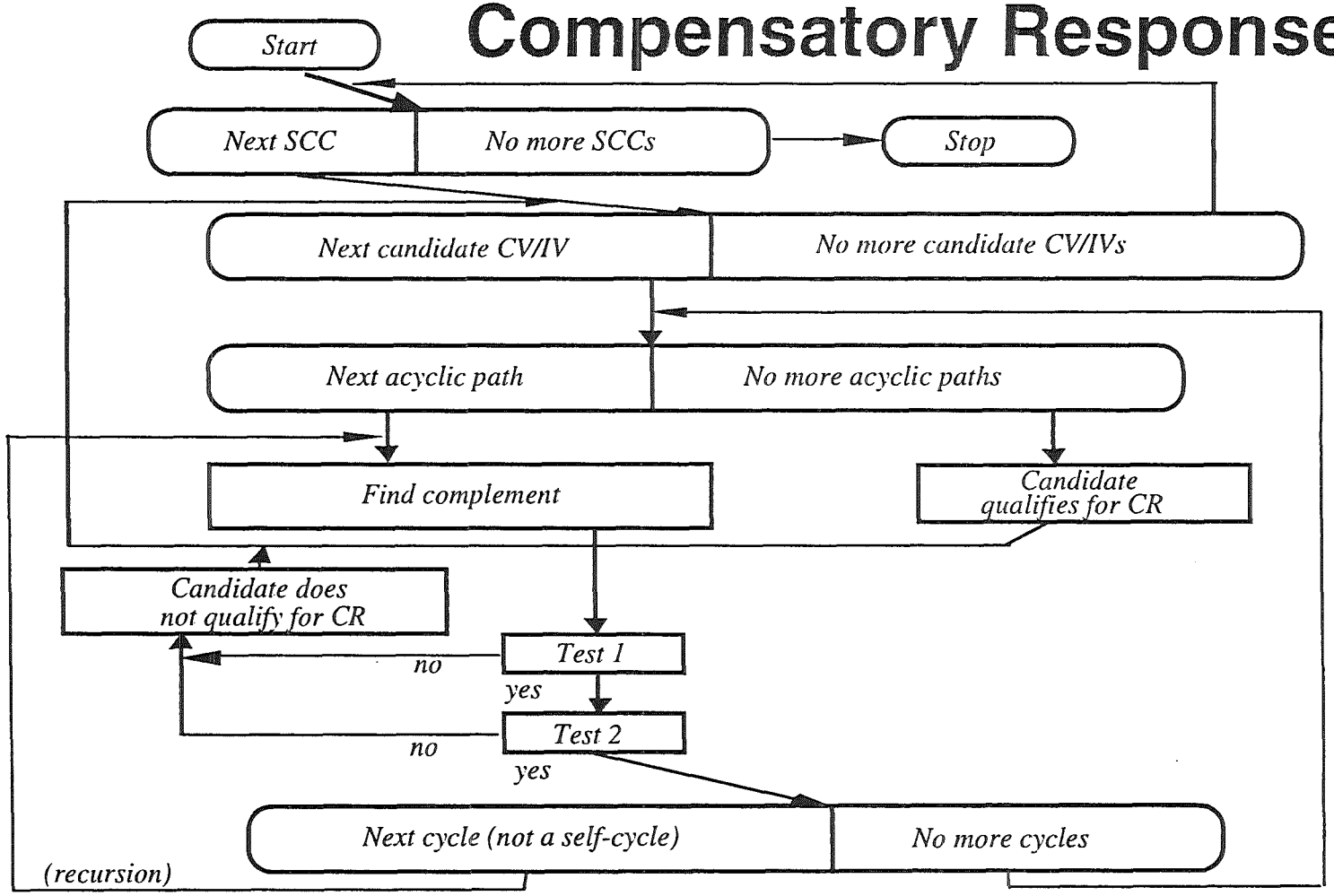


Qualitative Analysis of Causal Feedback

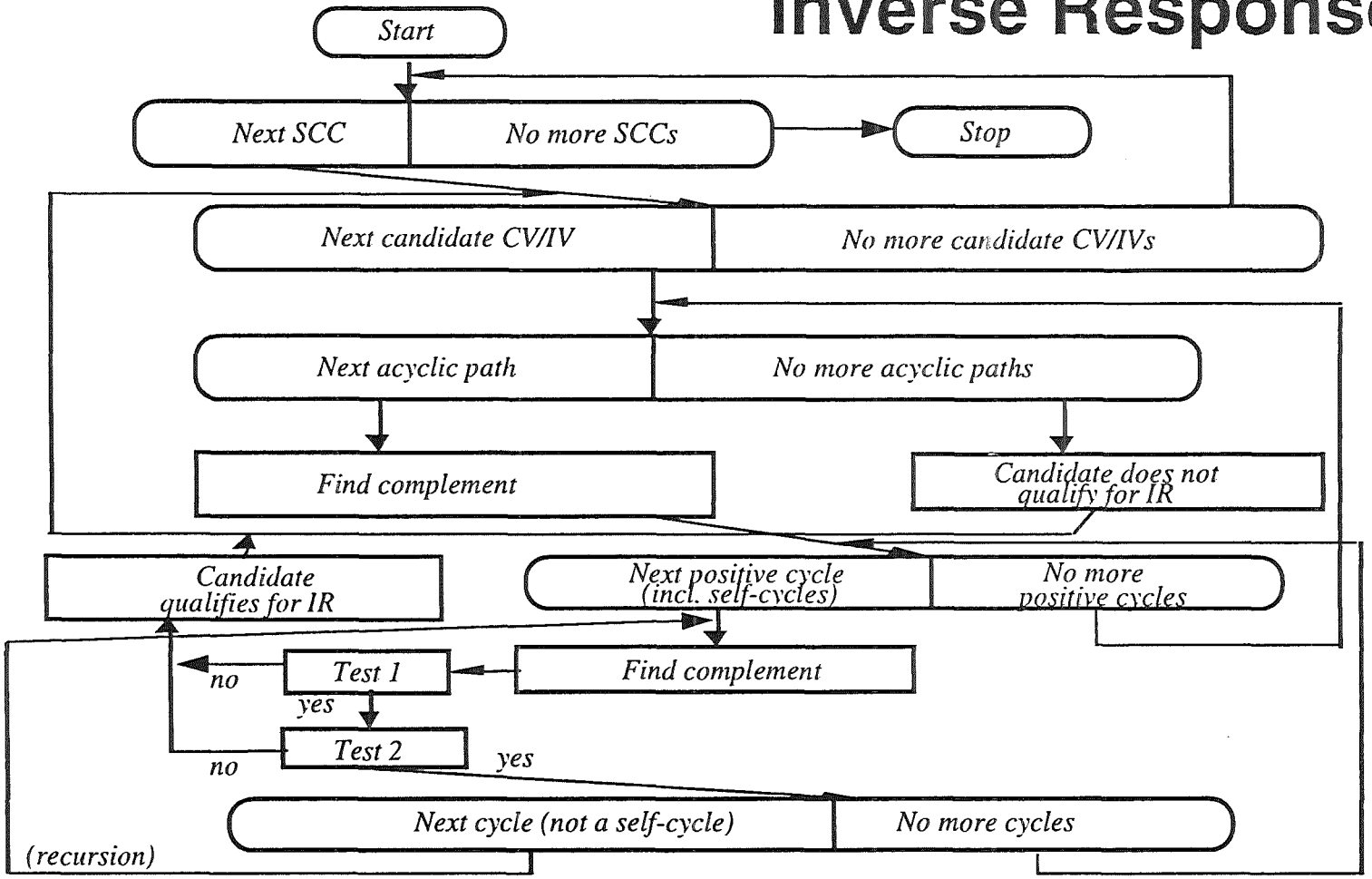
- *Starting Point: Signed Directed Graph*
- *Initial Response: Shortest Acyclic Path*
- *Ultimate Response:*
  - *Monotone (same as initial)*
  - *Compensatory*
  - *Inverse*

SIEMENS

Compensatory Response



Inverse Response



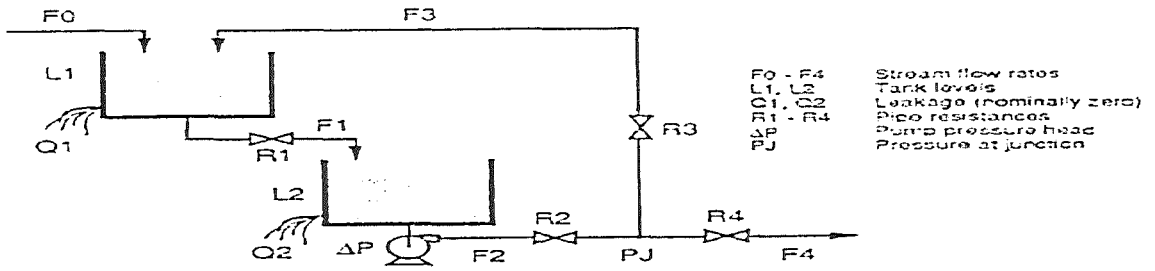
SIEMENS

Tests for Ultimate Response

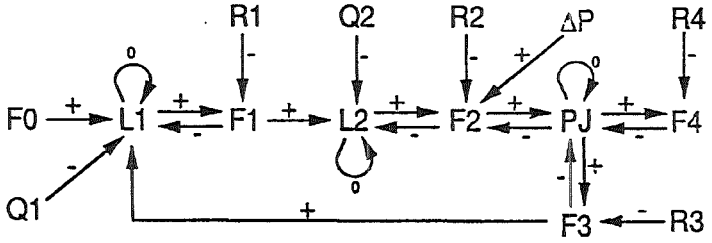
The complementary subsystem to each acyclic path from the disturbance to the state variable contains

Test 1: at least one zero self-cycle
Test 2: no cycle containing all variables of the sub-system

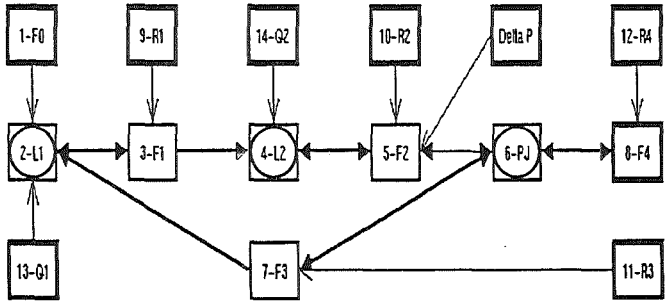
Recycle Tank System



System Model



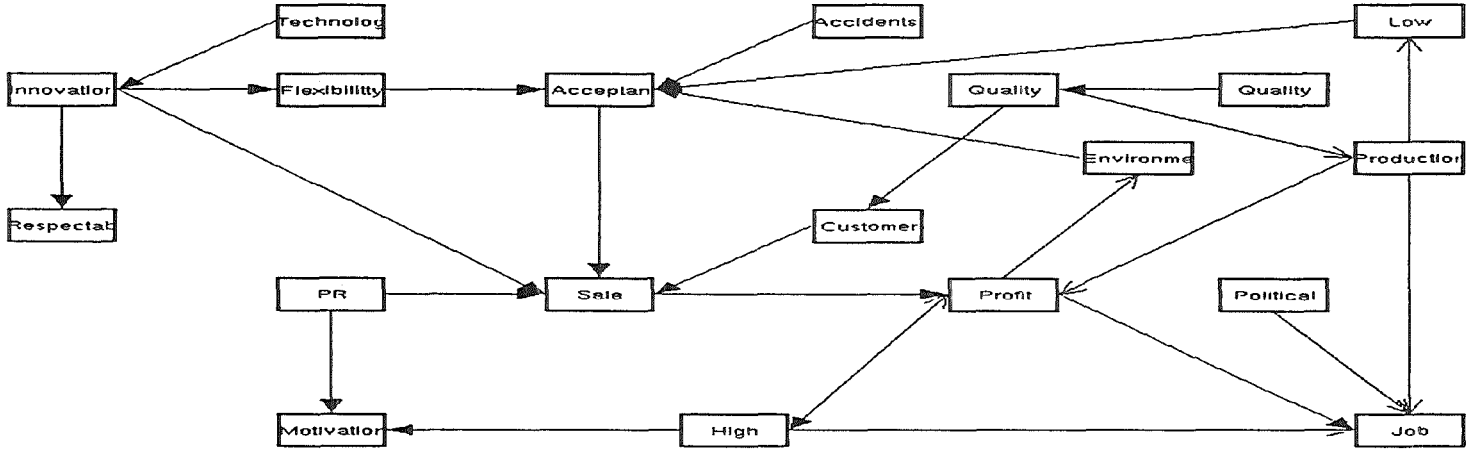
Signed Directed Graph SDG



Realization in CATS

SIEMENS

Example of a Socio-Technical System



For a deviation in disturbance a Stock(Innovation):
  In strongly connected component SCC(Main-3):
    state variable a Stock(Acceptance): intial -> '-' ultimate -> 'cv'
    state variable a Stock(Environment Amicability): intial -> '-' ultimate -> 'cv'
    state variable a Stock(High Wages): intial -> '+' ultimate -> same as initial
    state variable a Stock(Profit): intial -> '+' ultimate -> 'cv'
    state variable a Stock(Sale): intial -> '+' ultimate -> same as initial
  In strongly connected component SCC(nil-4):
    state variable a Stock(Flexibility): intial -> '+' ultimate -> '+'
  In strongly connected component SCC(nil-5):
    state variable a Stock(Respectability): intial -> '+' ultimate -> '+'

For a deviation in disturbance a Stock(Political Instabilities):
  In strongly connected component SCC(nil-1):
    state variable a Stock(Job Security): intial -> '-' ultimate -> '-'

For a deviation in disturbance a Stock(Technological Breakthru):
  In strongly connected component SCC(nil-6):
    state variable a Stock(Innovation): intial -> '+' ultimate -> '+'

For a deviation in disturbance a Stock(Low Prices):
  In strongly connected component SCC(Main-3):
    state variable a Stock(Acceptance): intial -> '+' ultimate -> 'cv'
    state variable a Stock(Environment Amicability): intial -> '-' ultimate -> 'cv'
    state variable a Stock(High Wages): intial -> '+' ultimate -> same as initial
    state variable a Stock(Profit): intial -> '+' ultimate -> 'cv'
    state variable a Stock(Sale): intial -> '+' ultimate -> same as initial

For a deviation in disturbance a Stock(Customer Satisfaction):
  In strongly connected component SCC(Main-3):
    state variable a Stock(Acceptance): intial -> '-' ultimate -> 'cv'
    state variable a Stock(Environment Amicability): intial -> '-' ultimate -> 'cv'
    state variable a Stock(High Wages): intial -> '+' ultimate -> same as initial
    state variable a Stock(Profit): intial -> '+' ultimate -> 'cv'
    state variable a Stock(Sale): intial -> '+' ultimate -> same as initial

For a deviation in disturbance a Stock(High Wages):
  In strongly connected component SCC(nil-2):
    state variable a Stock(Motivation of Employees): intial -> '+' ultimate -> '+'
  In strongly connected component SCC(nil-1):
    state variable a Stock(Job Security): intial -> '-' ultimate -> '-'

Hierarchischer Modellaufbau und Anwendung des Klassenkonzepts am Beispiel von RegioPlan$^+$

Dipl. Inf. Norbert Grebe, Universität Passau, 94030 Passau

1. Simulation in der Regionalplanung

RegioPlan$^+$ ist ein Modell, das den Einsatz der Simulation in der Regionalplanung ermöglicht. Es unterstützt die Entscheidungsträger bei der Analyse des realen Systems einer Region und bei der Entwicklung und Bewertung von planerischen Maßnahmen. Darüber hinaus werden Einheimische und Besucher durch das Experimentieren am Modell für die Problematik des landschaftlichen Gebiets sensibilisiert, indem sie verschiedene Szenarien am Rechner durchführen können.

Um das komplexe System einer Region verständlich im Modell abzubilden, setzt sich RegioPlan$^+$ aus vielen überschaubaren, selbständig lauffähigen Subkomponenten zusammen, die in vier Hierarchieebenen miteinander verbunden sind. Dabei ist mehrmals das Klassenkonzept des Simulators SIMPLEX II angewandt worden.

2. Der hierarchische Aufbau am Beispiel der Bevölkerung

Die Komponente Bevölkerung ist zunächst eine von elf Subkomponenten von RegioPlan$^+$ (siehe Abb. 2). Da sie im wesentlichen Arbeitskräfte für den regionalen Arbeitsmarkt zur Verfügung stellt, besteht sie ihrerseits aus fünf Teilkomponenten, die den Produktivkräften des Gesamtmodells entsprechen (Landwirtschaft, Forstwirtschaft, Produzierendes Gewerbe, Handel und Dienstleistungen und dem - wegen seiner Bedeutung in den Untersuchungsgebieten herausgelösten - Tourismusgewerbe) sowie einer Summenkomponente (siehe Abb. 3).

Der Bereich Tourismusgewerbe ist analog der zugehörigen Produktivkraft weiterhin in die Kategorien Kurhotels, Hotels gehobener Ausführung, Standardhotels und Parahotellerie unterteilt (siehe Abb. 4 Komponente TourisBv).

Die Bereiche Landwirtschaft bis Dienstleistungen und die vier Hotelkategorien lassen sich mit der gleichen Dynamikbeschreibung modellieren. Hier findet das Klassenkonzept von SIMPLEX II Anwendung. Abb. 5 zeigt die Struktur dieser Komponente mit der Bezeichnung BvSektor.

Eine weitere Komponente auf der untersten Hierarchieebene ist die Summenkomponente, die in Abb. 6 dargestellt ist.

3. Beschreibung der Komponenten

Die Struktur von RegioPlan$^+$ ist dergestalt, daß es sich durch geeignete Parametersätze auf verschiedene Regionen übertragen läßt. Die wesentlichen Komponenten und deren vielfach gegenseitigen (eventuell zeitverzögerten) Abhängigkeiten verdeutlicht der Modellaufbau in Abb. 2. Ein Pfeil symbolisiert die Übergabe eines Bündels von Größen, die in der Komponente an seinem Ursprung berechnet und von der Komponente an seinem Ende benötigt werden.

- Komponente Bevölkerung:

  Sie berechnet die Bevölkerungsentwicklung, die Zahl der Erwerbsfähigen und der Erwerbslosen sowie die Verschmutzung und den Verkehr, der durch die Bevölkerung verursacht wird.

- Komponente Besucher:

  Hier wird die Anzahl an Besuchern errechnet, die in die Region kommt. Dabei wird in fünf Nachfragetypen unterschieden: Gäste der Kurhotels, der gehobenen bzw. der Standardkategorie, der Parahotellerie sowie Tagesausflügler. Das Hauptunterscheidungsmerkmal bei den Übernachtungsgästen sind die Tagesausgaben. Wegen der strukturell gleichen Dynamik wurde bei der Implementierung in SIMPLEX-MDL[1] hier und in anderen Komponenten vom Klassenkonzept Gebrauch gemacht. Weiterhin ermittelt die Komponente das Kurtaxenaufkommen sowie den von den Besuchern verursachten Verkehr und die Verschmutzung.

- Komponente Produktivkräfte:

  Die Komponente Produktivkräfte besteht aus den Bausteinen für die Landwirtschaft (L), die Forstwirtschaft (F), das produzierende Gewerbe (G), Handel und Dienstleistungen (D) und für das Tourismusgewerbe (T). In jeder dieser unter Anwendung des Klassenkonzepts entstandenen fünf Subkomponenten findet im wesentlichen die Berechnung der Entwicklung des Anlagevermögens statt mit den daraus resultierenden Größen Ergebnis, genutzte Fläche und Erwerbstätige. Das Tourismusgewerbe unterscheidet analog zur Besucherkomponente das Übernachtungsangebot in vier verschiedenen Hotelkategorien. Die durch wirtschaftliche Aktivitäten verursachte Verschmutzung und Verkehr werden ebenfalls hier berechnet.

- Komponente Investition in Produktivkräfte:

  Hier finden neben der Berechnung der Steuerlast die Investitionsentscheidungen für die fünf Wirtschaftssektoren statt.

---

1. MDL = Model Description Language

- Komponente Infrastruktur:

  Diese Komponente ermittelt die Größe des Straßennetzes, der öffentlichen Infrastruktur (Schwimmbäder, Krankenhäuser etc.) und der Einrichtungen des Naturschutzes (Informationszentren, Waldlehrpfade etc.) mit ihrem zugehörigen Flächenbedarf. Da gut ausgebaute Infrastruktur- und Naturschutzeinrichtungen Besucher und Wohnbevölkerung anziehen, wird hierfür jeweils eine Attraktivität errechnet.

- Komponente Ausgaben für Infrastruktur:

  Hier findet die Ausgabenberechnung zum Erhalt und Ausbau der oben angeführten drei Bereiche statt. Außerdem werden Werbungsaufwand für die Region und Sozialausgaben der öffentlichen Hand ermittelt.

- Komponente Wohnung:

  Diese Komponente berechnet die Anzahl der Wohnungen für die Bevölkerung und die der Zweitwohnungen in der Region mit ihrem zugehörigen Flächenbedarf.

- Komponente Fläche:

  Sie besorgt die konsistente Aufteilung der Regionalfläche in die einzelnen Flächentypen (z.B. geschützte Fläche, landwirtschaftlich nutzbare Fläche, Siedlungs-, Gewerbe-, Wald-, Verkehrsfläche). Weiterhin wird wegen der Bedeutung für den Tourismus eine Attraktivität aus Landschaftsmix berechnet.

- Komponente Konsum:

  Diese Komponente berechnet die Nachfragen von Bevölkerung und Besuchern nach den Leistungen der fünf Wirtschaftssektoren.

- Komponente Außenwelt:

  Sie stellt die externen Einflüsse auf die untersuchte Region dar (z.B. Subventionen und Zuschüsse, Verschmutzung, Verkehr und externe Nachfragen und Attraktivitäten).

- Komponente Statistik:

  Sie ermittelt verschiedene statistische Gesamtgrößen des Modells wie z.B. Gesamtverschmutzung und -verkehr.

4. Beispiel einer Simulationsstudie

Für die Anwendung auf eine bestimmte Region wird RegioPlan$^+$ mit den für das Gebiet charakteristischen Werten vorbesetzt. Diese werden mit Hilfe eines standardisierten Fragebogens ermittelt. Struktur und Dynamik des Modells bleiben bei der Übertragung auf eine Zielregion unverändert.

Nach Eingabe der Startwerte für einige der Konstanten und der Zustandsvariablen steht dem Anwender ein leistungsfähiges Simulationsmodell zur Verfügung, das die Untersuchung verschiedener Szenarien ermöglicht.

Hierzu wird als erstes ein Referenzlauf durchgeführt, der die regionale Entwicklung ohne Eingriffe zeigt. Anschließend werden die für ein Szenario relevanten Modellgrößen geändert und die Simulation erneut angestoßen. Abb. 1 zeigt exemplarisch die Ergebnisse des Szenarios „Werbungsstopp“:

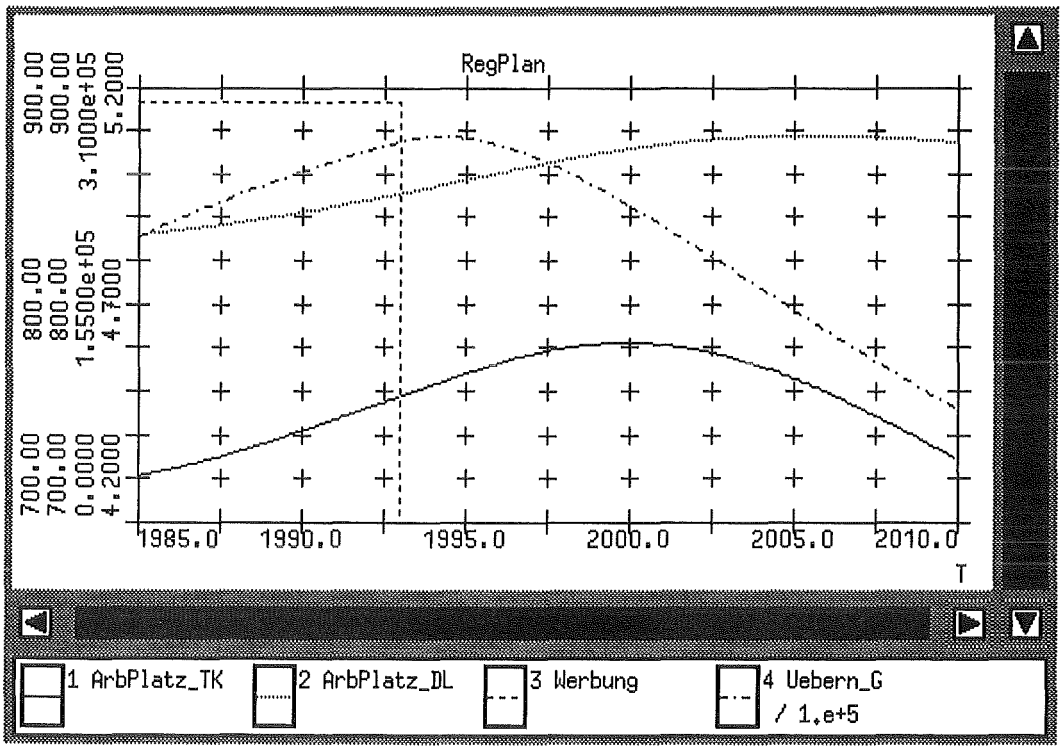


Abb. 1: Szenario Werbungsstopp

Dieses Szenario bedeutet, daß im Jahre 1993 die regionalen Werbemaßnahmen für den Tourismus vollständig eingestellt werden (Kurve 3). Mit einer kurzen Verzögerung sinken die Übernachtungszahlen in der gehobenen Hotelkategorie (Kurve 4). Dies gilt ebenfalls für die anderen Hotelklassen und die Zahl der Tagesausflügler. Daraufhin gehen mit unterschiedlichen Verzögerungszeiten die Arbeitsplätze im Tourismus (Kurve 1) und in Handel und Dienstleistung (Kurve 2) zurück.

5. Ausblick

RegioPlan^{+} besitzt - wie in Abb. 2 sichtbar - eine sehr komplexe Modellstruktur. Um das Arbeiten mit dem Modell zu erleichtern, erhält es im weiteren Verlauf der Entwicklung eine benutzerfreundliche graphische Oberfläche. Für die PC-Version ist bereits eine solche verwirklicht. Da RegioPlan^{+} beispielsweise zu Schulungszwecken oder zur Sensibilisierung von Touristen in Besucherzentren auch dem im Umgang mit Computern unerfahrenen Anwender zugänglich gemacht werden soll, ist bei der Gestaltung der Oberfläche vor allem auf Akzeptanz und Modellrobustheit zu achten.

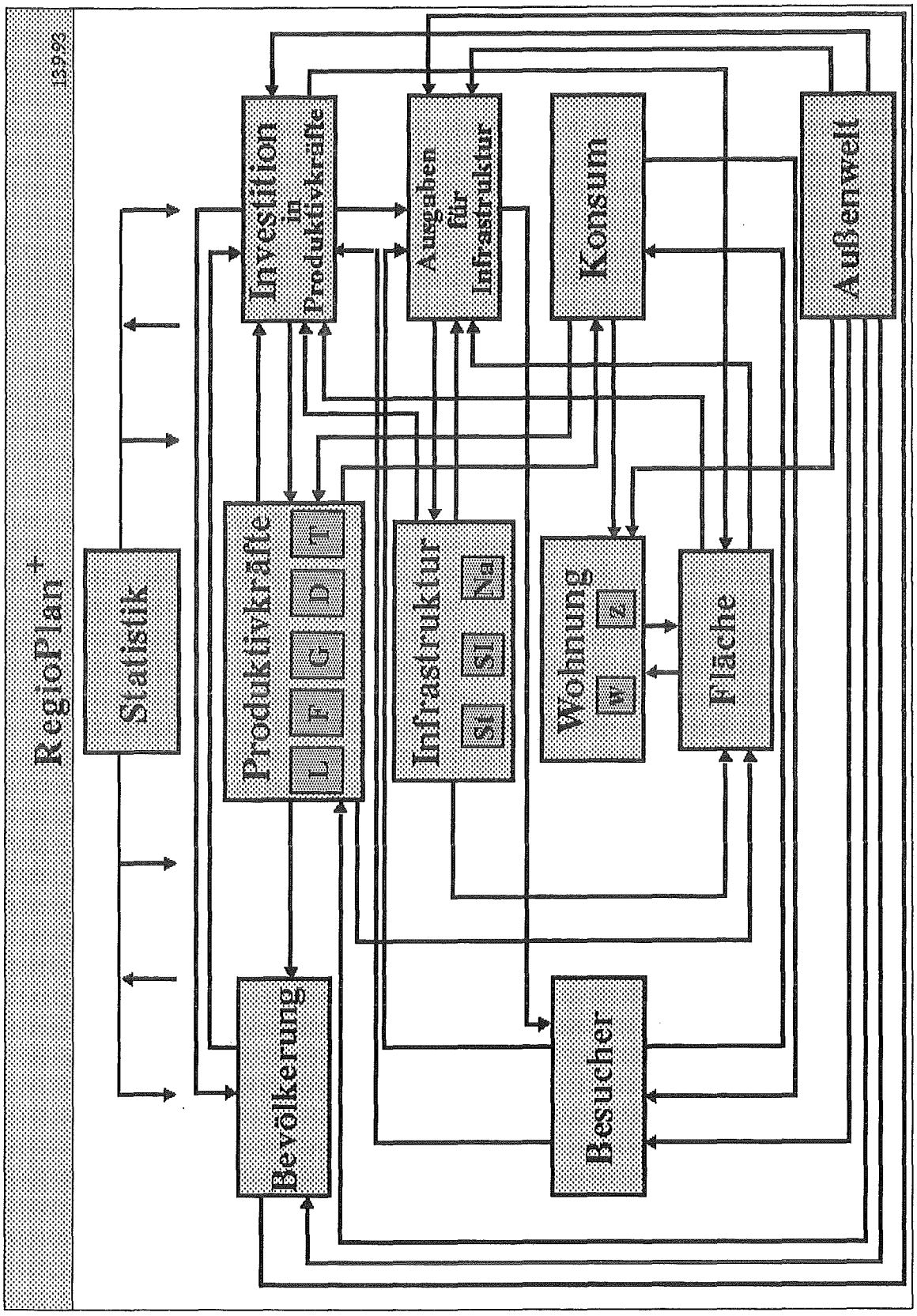


Abb. 2: Modellstruktur von RegioPlan^{+}

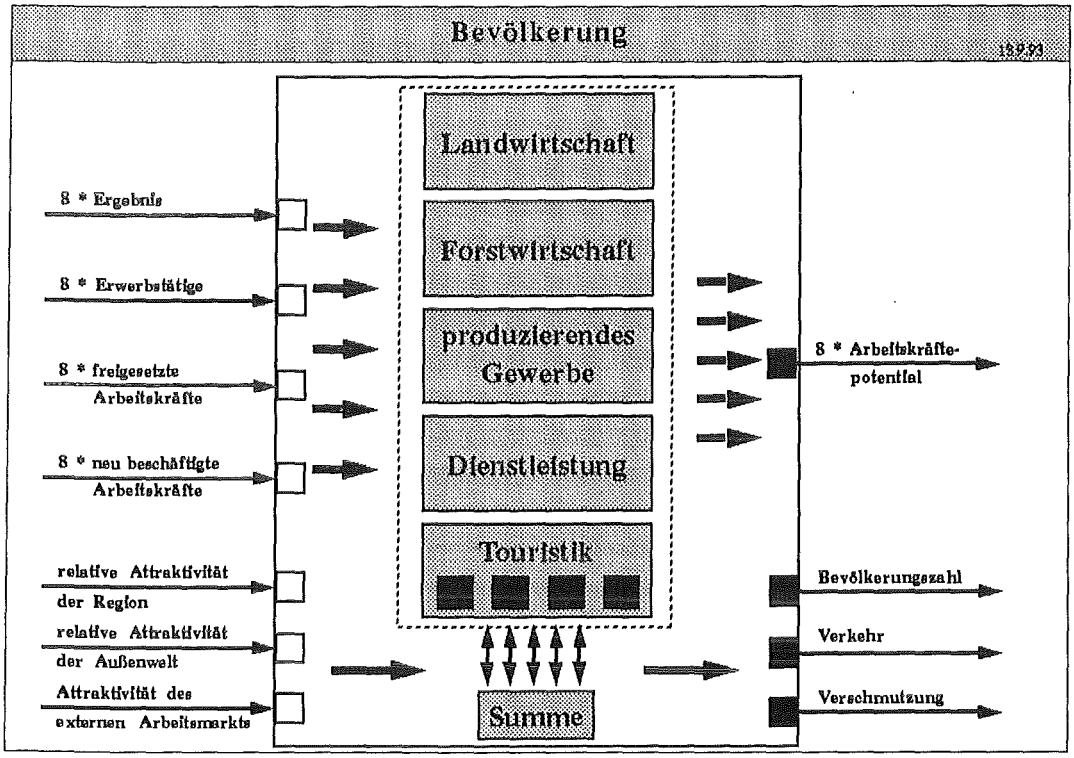


Abb. 3: Struktur des Teilmodells Bevölkerung

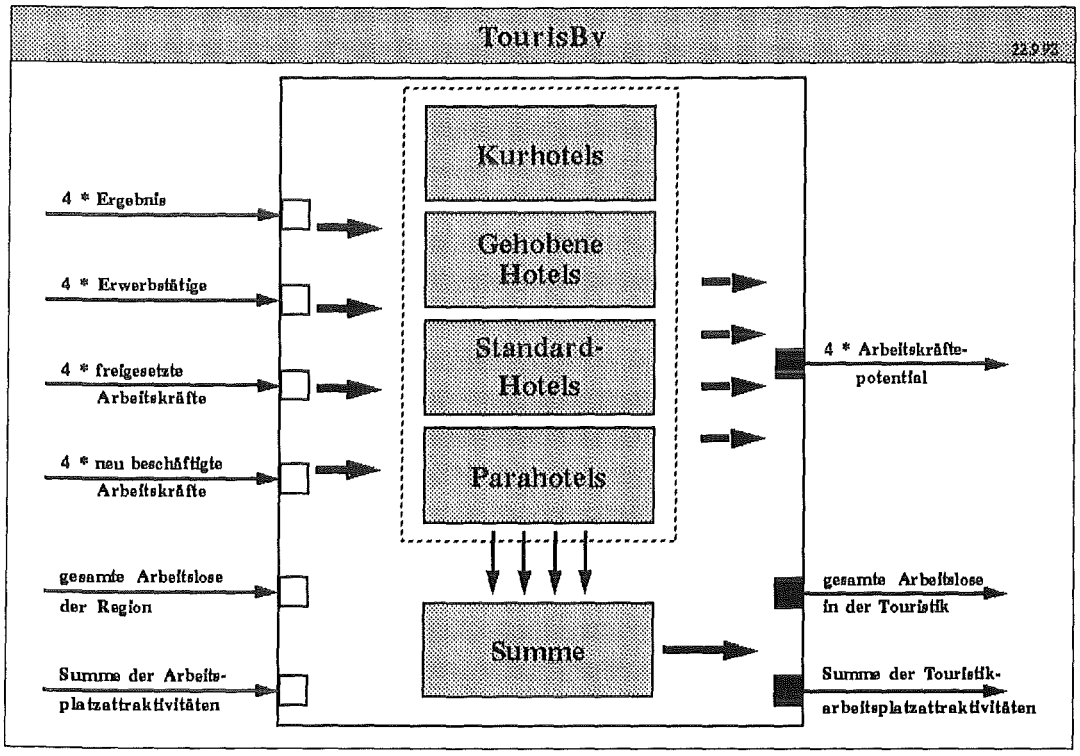


Abb. 4: Struktur des Teilmodells TourisBv

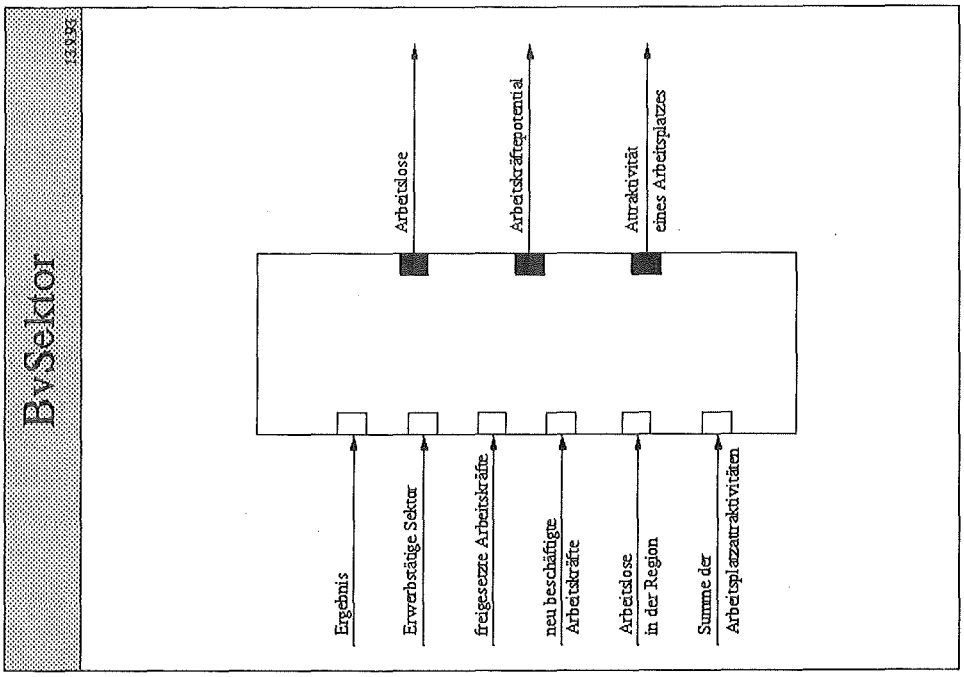


Abb. 5: Struktur von BvSektor

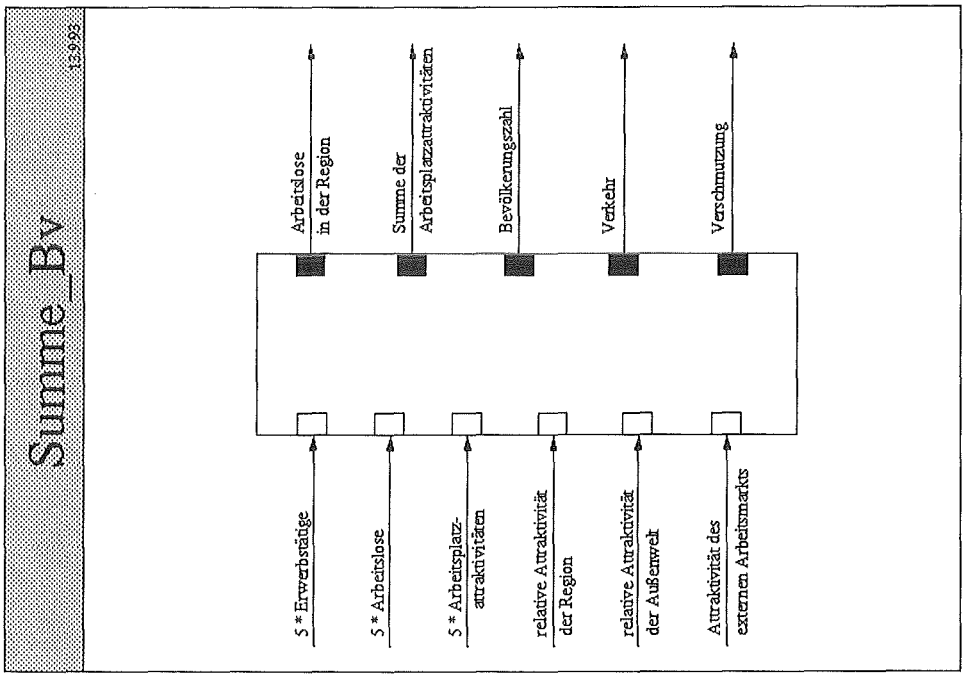


Abb. 6: Struktur von Summe_Bv

Ein neuronales Netz-Werkzeug zur Modellierung und Simulation dynamischer Systeme

Dr. Hubert B. Keller, Bernhard große Osterhues

Kernforschungszentrum Karlsruhe GmbH
Institut für Angewandte Informatik
Postfach 3640, 76021 Karlsruhe
Bundesrepublik Deutschland

Abstract:

Neuronale Netze finden aufgrund ihrer Lernfähigkeit zur Modellierung mathematisch schwer beschreibbarer Prozesse, wie sie besonders im Umweltbereich auftreten, wichtige Anwendungsmöglichkeiten. Am Beispiel der Müllverbrennung wird gezeigt, welche Einsatzgebiete für neuronale Netze konkret existieren. Die Eigenschaften verschiedener Netzarten werden beschrieben. Beispiele zum Lernen von Trajektorien zur Prozeßsimulation und die Beschreibung eines Werkzeugs erfolgen zum Schluß.

1 Einleitung

Im wesentlichen werden bei der Müllverbrennung die folgenden Ziele verfolgt:

- Reduktion der Müllmenge durch optimalen Ausbrand bei inertem Restmüll,
- energetisch optimale Verwertung sowie
- minimale Schadstofferzeugung.

Eine minimale Schadstofferzeugung bei gleichzeitig hoher Ausbrandgüte erfordert eine gleichmäßige Führung des Verbrennungsprozesses. Durch nichtlineare Zusammenhänge zwischen den Prozeßgrößen und den sich verändernden, a priori nicht vollständig bekannten, Müllzusammensetzungen existiert kein stationärer optimaler Zustandspunkt. Vielmehr existiert nur eine sich entsprechend der Müllzusammensetzung und -menge ändernde optimale Trajektorie, an welcher sich die Prozeßführung zu orientieren hat. Diese Trajektorie ist allerdings a priori nicht berechenbar und damit für vorher festgelegte konventionelle Regelalgorithmen nicht vorgebbar und einhaltbar.

Durch die Verwendung thermographischer Werte der Verbrennung unter Anwendung von Verfahren der *fuzzy logic* konnten Verbesserungen in der Regelung erzielt werden. Allerdings treten Probleme bzgl. der Wissensableitung und der Portierbarkeit auf. Die Wissensextraktion aus dem Erfahrungsschatz von Bedienern zur Ableitung von Regeln scheitert an Problemen wie intuitives, implizites, unvollständiges, vages Wissen (siehe z. B. /2/). Bei *fuzzy rules*, welche nur explitzites anlagenspezifisches a priori Wissen benutzen, kommt eine schlechte Übertragbarkeit auf andere Anlagen hinzu. Eine umfangreiche Systemanalyse ist bei jeder Installation unumgänglich, da ein *automatisches Einlernen* nicht möglich ist.
Allerdings haben diese Ansätze durch die Einbeziehung weniger Größen das Optimierungspotential in der Prozeßsteuerung von Müllverbrennungsanlagen deutlich aufgezeigt. Ausgeschöpft werden kann dieses Optimierungspotential nur durch Verfahren, welche neue Zusammenhänge selbsttätig ableiten können. Es sind also Verfahren einzusetzen, welche die Wirkzusammenhänge für Situationen erlernen und dann approximativ zur Verfügung stellen können. Dabei ist zu beachten, daß gerade auch die Vergangenheitswerte entsprechend der Durchlaufzeit im Prozeß als charakteristische Größen berücksichtigt werden müssen. Müllzusammensetzungen bzw. die daraus für den Verbrennungsprozeß und seiner Steuerung resultierenden Eigenschaften äußern sich in den Prozeßgrößen (örtlich verteilt) und deren zeitlichen Verläufe entsprechend der Durchlaufzeit des Prozesses. Diese Annahme ist die Grundvoraussetzung für alle Regelungs- / Steuerungskonzepte auf der Basis heuristischen Wissens und wird auch durch die Fähigkeit einzelner Bediener, den Prozeß näherungsweise optimal zu fahren, belegt. Neuronale Netze können die statistisch signifikant in Erscheinung tretenden Wirkzusammenhänge zwischen Eingangs- und Ausgangsgrößen für Steuerungszwecke lernen (z. B. Ausgangsgrößen im Prozeß und deren Abhängigkeiten von den Stellgrößen). Die Struktur des zu steurenden Prozesses spielt für die Regelung bzw. Steuerung eine große Rolle. Können relevante Parameter frühzeitig (d. h. zu Beginn der örtlich verteilten Regelstrecke) bestimmt werden, so ist nicht nur eine *feed forward*-Regelung, sondern auch die Adaption von Reglerparametern möglich. Im Bereich der Müllverbrennung können neuronale Netze eingesetzt werden, um

- Abhängigkeiten zwischen den Prozeßgrößen zu erkennen,
- den eingehenden Müll frühzeitig zu klassifizieren (Brennwert / Feuchte),
- nichtmeßbare Parameter zu schätzen,
- eine direkte Regelung durchzuführen (neuronaler Regler),
- Werte von thermographischen Messungen zu integrieren,
- einen Simulator zum Training der Anlagenfahrer oder zur Prognose von Systemzuständen zu realisieren.

2 Neuronale Netze

Im Vergleich mit *fuzzy logic* -Systemen bieten *Neuronale Netze* folgende Vorteile /4/:

- Fähigkeit zur Verarbeitung unstrukturierter Daten mit der Fähigkeit zur Abstraktion,
- Lernfähigkeit zur Erkennung von Beziehungen in einem dynamischen System,
- on-line Anpassung,
- Vereinfachung der Übertragbarkeit des erarbeiteten Wissens auf verschieden Anlagen durch Anpassung der Gewichtsparameter über Weiterlernen / Einlernen.

Für die genannten Anwendungen eignen sich z. B. Kohonen-Netze (*KN*) oder Backpropagationnetze (*BPN*). Kohonen-Netze haben die Fähigkeit, ähnliche Eingangsinformationskonstellationen in benachbarte Gebiete der Netzstruktur (hier 2D-Struktur) abzubilden. Eine Änderung der Eingangsinformationskonstellation ist somit als Wanderung der Aktivität im 2D-Netz zu erkennen. Ebenfalls kann hierdurch eine Segmentierung der Eingangsinformationen in 2D-Bereiche erfolgen. Backpropagation-Netze benötigen im Verhältnis zu Kohonen-Netze deutlich weniger Netzelemente, haben allerdings eine geringere Konvergenzgeschwindigkeit und es besteht die Gefahr des Übertrainierens (Fehler bei Interpolationen werden größer). Durch eine zweistufige Transformation kann ein 2D-Kohonennetz in einen linearen Vektor mit wahlfreiem Zugriff umstruktiriert werden, dies ist für eine Verarbeitung unter Echtzeitbedingungen wichtig.

2.1 Eigenschaften neuronaler Netze

Eine erste grobe Einteilung neuronaler Netze kann über die Architektur bzw. den internen Aufbau und ihrem operationalen Verhalten erfolgen.

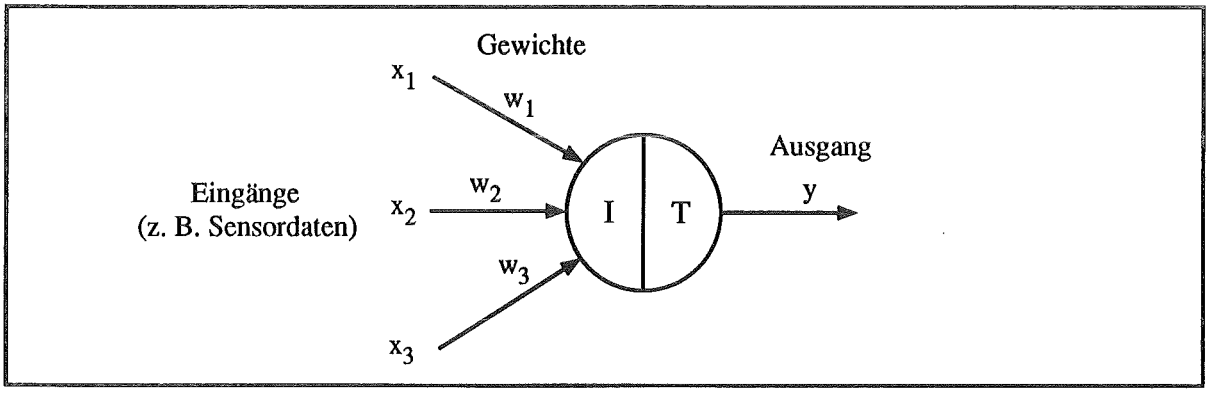
Unter dem Begriff der **Architektur** versteht man die Struktur des Netzes (Art der Verbindungen) und die Art der Bausteine, der Neuronen mit ihren internen Eigenschaften.

Das **operationale Verhalten** wird durch den Berechnungszeitpunkt für jedes Neuron (synchron / asynchron), die Informationsflußrichtung (vorwärts / rückgekoppelt (rekurrent)), das Vorhandensein eines internen Gedächtnis im Neuron (Abhängigkeit der Reaktion auf Eingangssignale von der Vergangenheit), das Lernprinzip (überwacht / nicht-überwacht) und das Lernproblem, wie diskrete Klassifikation bzw. approximative Berechnung (inklusive Mustervervollständigung), bestimmt.

2.1.1 Neuronen

Die Neuronen eines neuronalen Netzes, als Modellvorstellung für biologische Neuronen, werden häufig als Neuronen-Elemente, Prozeßelemente oder auch einfach als Units bezeichnet. Eine Neuron besteht aus einer Eingangs,- einer Aktivierungs- und einer Ausgangsfunktion. Die Ausgänge eines Neurons sind mit den Eingängen anderer Neuronen verbunden. Die Verbindungen (Synapsen) eines Neurons nehmen eine Aktivierung x_i mit einer bestimmten Stärke w_i von anderen Neuronen auf, summieren diese und lassen dann

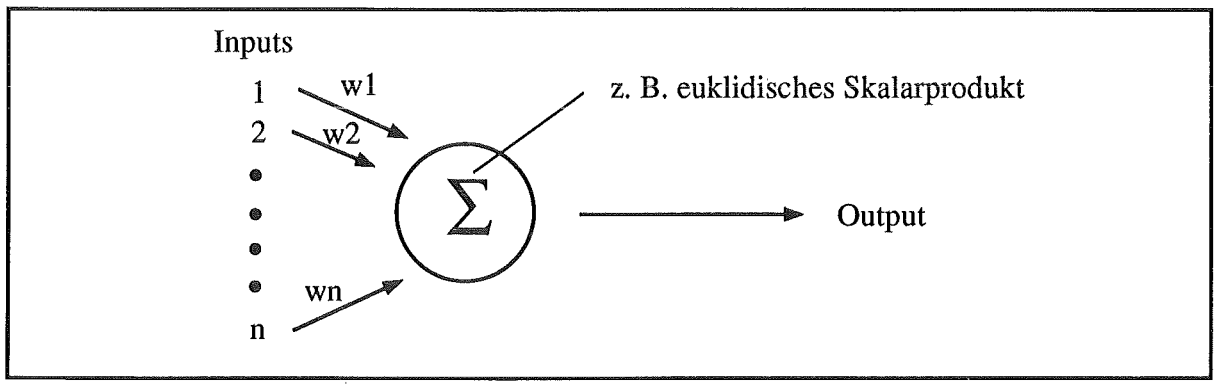
am Ausgang y (Axon) des Neurons eine Aktivität entstehen, sofern die Summe vorher einen Schwellenwert T überschritten hat /1, 5, 8/.



Das Grundmodell eines Neurons stützt sich im wesentlichen auf die Vereinfachung von McCulloch und Pitts aus dem Jahre 1943, die ein Neuron als **Addierer mit Schwellenwert** betrachten (mehrere Eingänge, aber nur einen Ausgang).
Durch die Verwendung unterschiedlicher Funktionen zur Summation der Eingänge (z. B. euklidisches Skalarprodukt, gewichtete Summe über das Produkt der Eingaben, Hamming-Distanz etc.), zur Aktivierung eines Neurons und zur Berechnung der Ausgabe des Neurons (z. B. binär, linear, linear-begrenzt oder sigmoid) können verschiedene Formen von Neuronen gebildet werden.

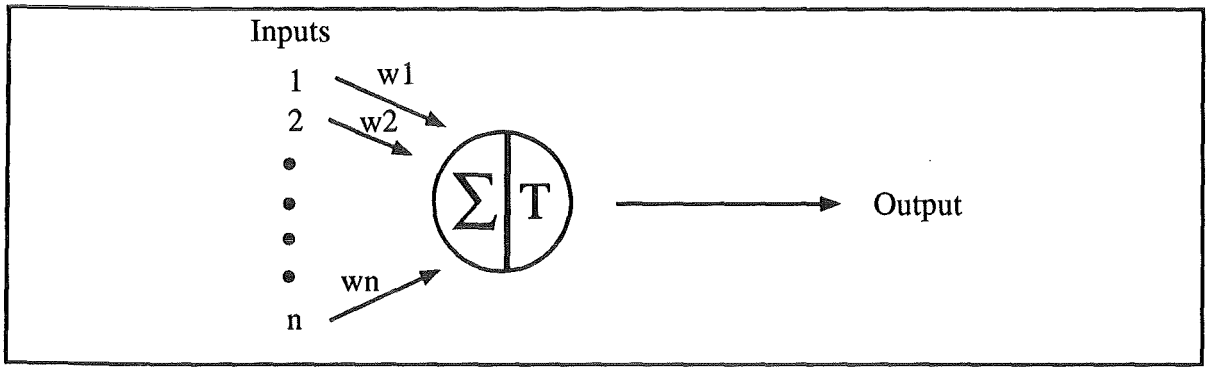
Die Dynamik eines neuronalen Netzes wird durch die Prozesse der Summationsfunktion (I), der Aktivierungsfunktion (A) und der Übertragungsfunktion (T) bestimmt.

Mathematisch kann man die Input-Signale und die Gewichte als Vektoren (i_1, i_2, ..., i_n) und (w_1, w_2, ..., w_n) auffassen. Die Berechnung des gesamten Input-Signals erfolgt mit Hilfe einer **Summationsfunktion**, z. B. die gewichtete Summe, d. h. dem euklidischen Skalarprodukt der beiden Vektoren, was geometrisch als ein Maß für die Ähnlichkeit dieser beiden Vektoren aufgefaßt werden kann.



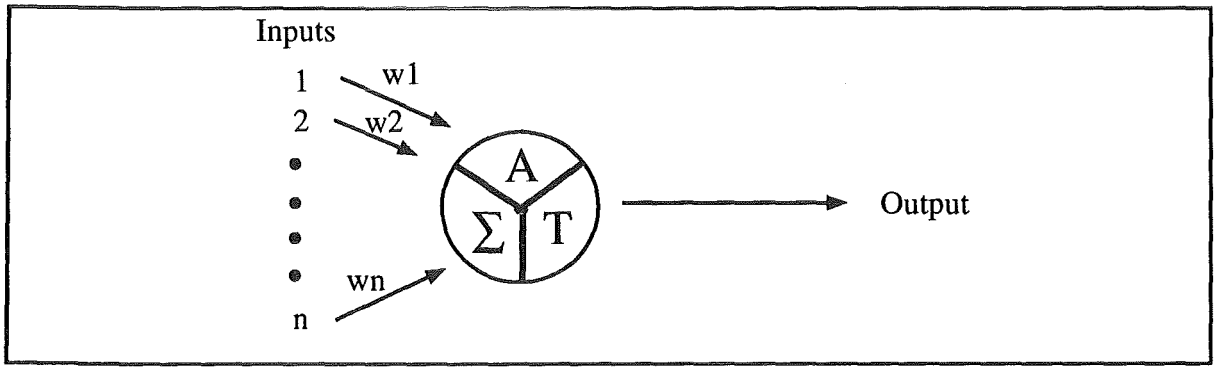
Eine wichtige Funktion ist die **Übertragungsfunktion** T (**Schwellenwert-, Output-Funktion**). Diese Funktion ist im allgemeinen nichtlinear, da lineare Funktionen aufgrund der Tatsache, daß ihr Resultat proportional zur Eingabe ist, nur beschränkte Möglichkeiten bieten und sich viele Probleme mit linearen Funktionen nicht lösen lassen.

Wenn die gewichtete Summe der Input-Signale größer als der Schwellenwert ist, dann erzeugt das Prozeßelement ein Signal. Falls das Ergebnis der Summationsfunktion kleiner als der Schwellenwert ist, dann wird kein Signal (oder ein hemmendes Signal) erzeugt; beide Antworten sind signifikant.



Die normalerweise üblichen Übertragungsfunktionen sind Treppenfunktionen oder sigmoide Funktionen, wobei in den Anwendungen sigmoide, d. h. S-förmige, Funktionen bevorzugt werden, da sie außer der Nichtlinearität noch über gewisse Differenzierbarkeitseigenschaften verfügen. Die Anforderungen an eine Übertragungsfunktion bestehen darin, daß diese für negative oder zu kleine positive Werte ein definiertes Minimum (entsprechend der Ruhefrequenz beim Neuron) oder 0 liefert und ab einem gewissen Schwellenwert allmählich oder abrupt einen Maximalwert, häufig 1, annimmt /5/.

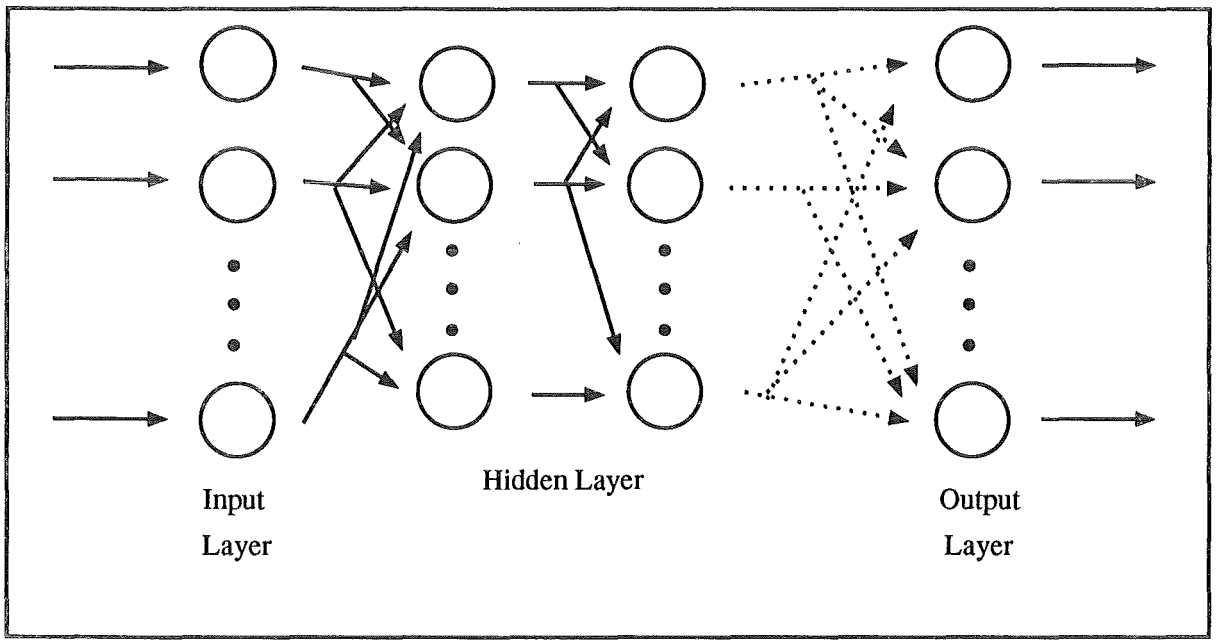
Ein weiterer Prozeß im Prozeßelement betrifft die **Aktivierungsfunktion**. Das Ergebnis der Summationsfunktion kann als Argument für eine Aktivierungsfunktion dienen, bevor es an die Übertragungsfunktion übergeben wird. Der Sinn der Aktivierungsfunktion besteht darin, daß sie es ermöglicht, die Ausgabe in Abhängigkeit von der Zeit zu variieren. Die Aktivierungen früherer Zeitpunkte geben dem Prozeßelement ein Gedächtnis, mit dem beispielsweise Adaption modelliert werden kann. Allgemein ist die Aktivierungsfunktion von vorhergehenden Aktivierungen und von einem Satz von Parametern abhängig /5/. In den meisten Fällen wird als Aktivierungsfunktion die Identität verwendet, da die Untersuchungen auf diesem Gebiet noch nicht sehr weit fortgeschritten sind. Den Wert der Aktivierungsfunktion bezeichnet man als **Aktivierungswert**.



2.1.2 Netzstruktur

Zur Strukturierung eines neuronalen Netzes werden Neuronen mit gleicher (paralleler) Funktion zu einem Funktionsblock (Schicht, Layer) zusammengefaßt. Die einzelnen Schichten werden in einer Folge "nacheinander" angeordnet und in geeigneter Weise miteinander verbunden. Jede Schicht ist wie ein Programm-Modul, das mit Hilfe der genau festgelegten Schnittstelle (Eingänge, Ausgänge, etc.) eine festgelegte Spezifikation erfüllt, wobei die eigentliche Implementierung nicht entscheidend ist, solange die gewünschte Funktion tatsächlich erbracht wird.

Schichten, die weder aus Eingabe- noch aus Ausgabeeinheiten bestehen, werden als verdeckte Schichten (hidden Layer) bezeichnet; das Netz in der folgenden Abbildung besteht beispielsweise aus drei Schichten (zwei mit vollständig vernetzten Eingängen, eine davon als verdeckte Schicht).



Die erste Schicht enthält die Eingabeeinheiten und dient ausschließlich der Verteilung der Eingabesignale; sie stellt eigentlich keine "echte" Schicht dar. Aufgrund der fehlenden Verbindungen zu den Ein- bzw. Ausgängen kann der Aktivitätszustand der Neuronen der verdeckten Schicht nicht direkt durch die "Außenwelt" festgelegt werden, sondern ist nur mittelbar über die internen Verbindungen des neuronalen Netzes beeinflußbar /1/. Die Neuronen einer verdeckten Schicht stellen die Grundlage zur Bildung neuer Merkmale und interner Repräsentationen dar /8/.

Für die Anzahl und die Richtungen der Verbindungen innerhalb eines neuronalen Netzes sind theoretisch nur kombinatorische Grenzen gesetzt. Für die Richtung der Verbindungen und damit des Informationsflusses kann man grundsätzlich die folgenden Formen unterscheiden:

- **Feed-forward (vorwärts gekoppelte) Netze:**
  Die Eingänge einer Schicht werden ausschließlich von den Ausgängen der Schicht davor gespeist, d. h. die Neuronen einer Schicht haben keinerlei Einfluß auf einander

und auf davorliegende Schichten. Die Grundidee dieser Charakterisierung besteht in der Existenz von Funktionseinheiten (Neuronen oder Neuronen-Schichten), die nicht rückgekoppelt sind.

Graphentheoretisch bedeutet dies, daß der zugehörige gewichtete, gerichtete Graph zyklenfrei ist.

- **Rekurrente Netze:**
  Rekurrente Netze werden auch **feed-back (rückgekoppelte) Netze** genannt. In ihnen kann jedes Neuron mit sich oder mit jedem anderen Neuron des Netzes verbunden sein. Es handelt sich somit um Netzwerke, die Verbindungen und damit eine Informationsverarbeitung in alle Richtungen (d. h. auch Querverbindungen und Rückkopplungen) zulassen. Aus diesem Grund haben rekurrente Netze in der Regel keine Schichtenstruktur. Um ein Ergebnis zu erhalten / produzieren muß ein rekurrentes Netz dieses wiederholt berechnen, bis die Neuronen einen stabilen Zustand erreichen.

  Graphentheoretisch bedeutet dies, daß der zugehörige gewichtete, gerichtete Graph auch Zyklen enthalten kann.

2.2 Back-Propagation-Netze

Der Back-Propagation Algorithmus eignet sich als Lernverfahren für überwachtes Lernen bei multilayer Netzwerken. Die grundlegende Idee des Back-Propagation-Algorithmus liegt in der Verbindung eines neuronalen Netzes auf der Basis von Perceptrons /5, 8/ mit einer (nicht-linearen) Schwellenwertfunktion und einem Gradientenverfahren. Der Back-Propagation-Algorithmus ist immer dann einsetzbar, wenn ein neuronales Netz mit Beispielen trainiert werden soll, um eine unbekannte stetige Funktion $y = f(x)$ möglichst gut zu approximieren.

Back-Propagation ist eine Verallgemeinerung der Delta-Regel auf Netzwerke mit beliebig vielen Schichten und einer feed-forward-Verarbeitung. Im Gegensatz zum Perceptron enthält jedes Neuron nun eine kontinuierliche sigmoide Ausgabefunktion mit Werten aus dem Intervall [0, 1]. Eine solche Funktion ist beispielsweise die Fermifunktion /7/:

$$\sigma(x) = (1 + \exp(x))^{-1}.$$

Die prinzipielle Idee ist, daß die verdeckten Neuronen eine interne Repräsentation der Musterassoziationen durch Rückwärtspropagieren eines Fehlers von der Ausgabeschicht in Richtung Eingabeschicht (error-back-propagation) erlernen. Die grundlegende Idee des Fehlerrücksendens wurde von Rosenblatt bereits 1962 formuliert.

Die Ausbreitung durch das Netz erfolgt schichtweise, d. h. die Aktivierungen der Neuronen werden zuerst in jener verdeckten Schicht berechnet, die der Eingabeschicht am nächsten liegt. Dann erfolgt die Berechnung der Aktivierung der nächsten, näher bei der Ausgabeschicht gelegenen Schicht, bis schließlich die Ausgabeschicht selbst erreicht wird.

In der zweiten Phase erfolgt die Fehlerbestimmung und die entsprechende Gewichtsveränderung, wiederum schichtweise. Dabei wird bei der Ausgabeschicht begonnen, da hier das gewünschte Muster zur Verfügung steht (überwachtes Lernen) und mit dem tatsächlichen vom Netz produzierten Muster verglichen werden kann. Aus der Differenz dieser beiden Muster wird ein sogenanntes Fehlersignal gebildet, von dem einerseits die Änderung der Gewichte zwischen der Ausgabeschicht und dessen benachbarter verdeckter Schicht und andererseits auch die Berechnung des neuen Fehlersignals für die nächste verdeckte Schicht abhängt.

Wurde der Fehler bis zur letzten verdeckten Schicht zurückgesendet und wurden dabei alle Gewichtsänderungen vorgenommen, dann kann ein (neues) Muster angelegt und vorwärts propagiert werden. Wendet man diese Lernprozedur wiederholt an, so wird der Fehler schrittweise verringert. Wichtig ist, daß man andere, bereits erlernte Muster mit diesem Vorgehen wieder zerstören kann (Übertrainieren). Der Lernvorgang wird, wenn der Fehler entsprechend klein geworden ist, als beendet angesehen und abgebrochen.

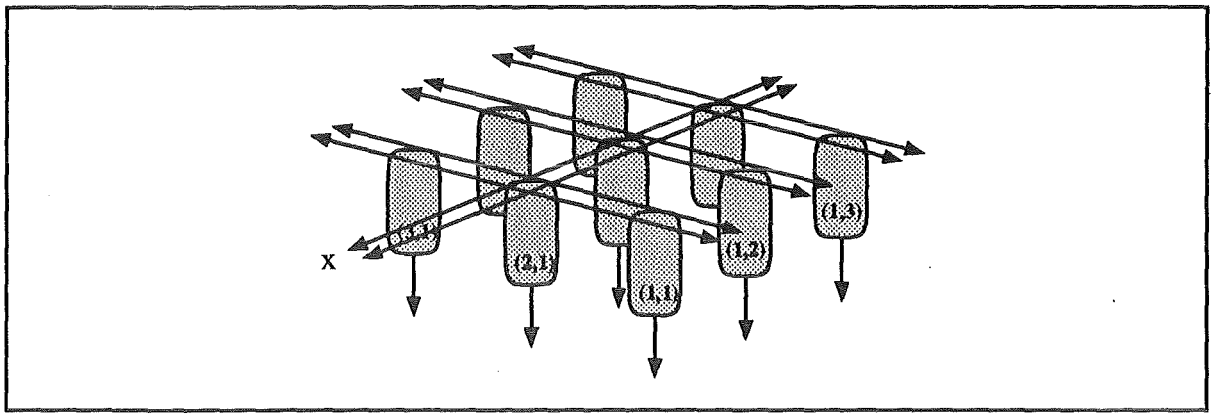
Da es sich bei dem Back-Propagation-Algorithmus um ein Gradientenverfahren handelt, kann sich das neuronale Netz in einem lokalen Minimum verfangen und die gestellte Aufgabe nicht fehlerfrei erlernen (d. h. der "Algorithmus konvergiert nicht gegen die richtige Netzwerkantwort"). Bei binären Aufgabenstellungen ist das nicht störend, da die Entscheidbarkeit (und nicht die Fehlerfreiheit) ausschlaggebend ist.

2.3 Kohonen-Netze

2.3.1 Allgemeine Eigenschaften

Eine wichtige Wechselwirkung zwischen Neuronen ist die gegenseitige Hemmung der Neuronen innerhalb einer neuronalen Funktionsgruppe (Schicht). Indem man allen Neuronen regelmäßig angeordnete Punkte eines Raumes (Koordinaten) zuordnet, werden zwischen den Neuronen Abstände und Nachbarschaften definiert. Die Wechselwirkungen innerhalb einer Schicht werden zu lokalen Wechselwirkungen, die bei verschiedenen Mitgliedern dieser Schicht verschieden wirken können. Diese Anordnung bietet besondere Möglichkeiten: Sie kann nicht nur eine Abbildung von hochdimensionalen Musterpunkten auf niederdimensionale (z. B. zweidimensionale Gitter-) Punkte bewirken, sondern dabei auch die wichtige Eigenschaft besitzen, daß benachbarte Muster auch auf benachbarte Neuronen abgebildet werden. Eine Abbildung, die den Musterraum reduziert, verliert dabei zwar Information; um jedoch die Probleme bei der Trennung hochdimensionaler Musterklassen weiterhin zu verdeutlichen, sollte eine solche Abbildung die wichtige Eigenschaft besitzen, daß benachbarte Punkte im Musterraum auch im Bildraum benachbart bleiben. Abbildungen mit der Eigenschaft der Nachbarschaftserhaltung werden in der Mathematik auch topologie-erhaltend genannt /6, 7, 8/. Es seien N Punkte $w_1, \ldots, w_n$ gegeben, die beispielsweise in einer zwei-dimensionalen Nachbarschaft zusammenhängen sollen. Assoziiert man mit jedem Punkt (Gewichtsvektor) eine Prozessoreinheit (Neuron), so hat bei einer einfachen 2-dimensionalen Netzstruktur jedes Neuron vier direkte Nachbarn. Als Eingabe erhalten alle Neuronen parallel die gesamte Eingabeinformation des

n-dimensionalen Mustervektors x (vgl. folgende Abbildung aus /1/).



Als Mittelpunkt einer Aktivierung kann nun ein einziges Neuron c mit den Koordinaten (i,j), als "selektiertes Neuron" mit der stärksten Aktivierung identifziert werden (winner-takes-all Prinzip). Das Gewicht des Neurons c und die Gewichte der "umliegenden" Neuronen werden durch eine Funktion modifiziert, die dem Querschnitt eines Sombrero gleicht und daher **Mexikanerhutfunktion** genannt wird. Durch die Mexikanerhutfunktion werden die Gewichte der Neuronen des Neuronengitters in Abhängigkeit von der Entfernung zu c verändert; d. h. die Gewichte der Neuronen die sehr nahe bei c liegen, werden erhöht, bei entfernteren Neuronen werden sie abgeschwächt. Daher ist nicht nur das Neuron c, sondern parallel dazu alle Neuronen k innerhalb der Nachbarschaft am Lernprozeß beteiligt. Den Konkurrenzkampf um die höchste Aktivierung gewinnt aufgrund des Prinzips der Nachbarschaftserhaltung nur dasjenige Neuron c, dessen Gewichtsvektor den kleinsten Abstand zum Eingabemuster x hat. Diese Form der Aktivierung bedeutet eine Quantisierung des Eingabemusters (**Vektorquantisierung**): Alle geringfügigen Variationen dieses Musters werden auf ein und denselben Gewichtsvektor abgebildet. Aufgrund dieser Vorgehensweise wird eine Aufteilung des Musterraumes in einzelne Klassen vorgenommen; der Gewichtsvektor ist der Klassenprototyp.

Im Zuge der Lernphase werden zum einen die Gewichtsvektoren benachbarter Neuronen in die gleiche Richtung korrigiert, so daß sie die Tendenz haben ähnlich zu werden (**Selbstordnung**); zum anderen erhalten die Neuronen der Gitterrandpunkte als Eingabe mindestens die Punkte x am Rande der Verteilung, so daß ihre Gewichtsvektoren "an den Rand streben" und das Netz aus den Gewichtsvektoren im Laufe der Zeit auseinanderziehen (**Entfaltung**). Diese von Teuvo Kohonen entwickelte Lernregel wurde durch das Lernen in biologischen Systemen inspiriert und wurde ursprünglich nur in nicht-überwachten Lernsituationen angewandt. Entsprechende Erweiterungen erlauben die Anwendung auf überwachte Lernprobleme (siehe z. B. /7/). Das im folgenden Abschnitt beschriebene Verfahren ist auf überwachte Lernprobleme anwendbar.

2.3.2 Lernalgorithmus

Voraussetzung für ein gutes Lernverhalten eines Kohonen-Netzes ist die zugrunde liegende Datenbasis. Die Datenbasis sollte sehr zuverlässig sein und das zu 'lernende System' möglichst genau beschreiben. D. h. alles was das neuronale Netz lernen soll, muß in der Datenbasis in irgendeiner Form vorhanden sein.

Die Datenbasis stellt die Netzeingabe dar und wird durch eine Menge L von Lernbeispielen repräsentiert:

$$L = \{L_1, L_2, \ldots, L_B\}$$

mit $L_i = (E_i, A_i)$ und mit $E_i = (e_1, \ldots, e_l)$, bzw. $A_i = (a_1, \ldots, a_m)$.

Jedes Lernbeispiel L_i besteht aus einem Eingabevektor E_i und dem dazugehörigen Ausgabevektor A_i (überwachtes Lernen). Das Lernbeispiel stellt dabei den Systemzustand zu einem bestimmten Zeitpunkt t dar. die Größen in den Vektoren können z. B. der Prozeßinput (z. B. Stoffmengen) oder der Prozeßoutput als gemessene bzw. abgeleitete Zustände sein.

Der Algorithmus für das Lernen in Kohonen-Netzen sieht folgendermaßen aus:

(1) Initialisierung

- Lernschrittzahl t_{max} , Anzahl der Neuronen und Dimensionen von Ein- und Ausgabevektor festlegen
- ε_{max} , ε_{min} , δ_{max} ,δ_{min} festlegen
- Alle Gewichte wr_{in} und wr_{out} mit Zufallszahlen belegen

(2) Zufällige Auswahl eines Lernvektors L_i (sensorisches Signal) aus der Menge L aller Lernbeispiele

(3) Bestimmung des Erregungszentrums, d. h. das Neuron s für das gilt:

$$\|w_s - E_i\| = \min^N_{r=1} \{\|w_r - E_i\|\}$$

$\|E\|$: Norm eines Vekrors E
w_s : Gewichtsvektor des Neurons s (Erregungszentrum)

(4) Lernschritt für Eingabevektoren E

$$w_{r,in}(t+1) = w_{r,in}(t) + \varepsilon * h(r,s,\delta) * (E - w_{r,in}(t)), \qquad \text{für alle Neuronen r}$$

(5) Lernschritt für Ausgabevektoren A

$$w_{r,out}(t+1) = w_{r,out}(t) + \varepsilon * h(r,s,\delta) * (A - w_{r,out}(t)), \qquad \text{für alle Neuronen r}$$

(6) Gehe zu Schritt (2), solange bis t_{max} erreicht ist.

In **Schritt 1** des Algorithmus werden die folgenden Größen festgelegt:

(1) Lernschrittzahl

Die Lernschrittzahl legt fest, wie oft das Lernverfahren durchlaufen werden soll.

(2) Größe des Netzes

Die Größe des Netzes wird durch die Anzahl der Neuronen bestimmt. Im allgemeinen kann man davon ausgehen, daß ein Netz bei sonst gleichen Bedingungen mit einer größeren Anzahl von Neuronen besser 'lernt' als mit einer geringeren.

(3) Dimension der Ein- und Ausgabe

Die Dimension der Eingabe legt die Anzahl der Größen fest, auf die das Netz trainiert werden soll. Durch die Dimension der Ausgabe wird die Anzahl der Größen bestimmt, die das Netz ausgeben soll.

(4) Lernrate ε

ε legt die maximale Lernschrittweite fest. ε sollte mit der Anzahl der Lernschritte vom Anfangswert ε_{max} zum Endwert ε_{min} hin abnehmen, um die Konvergenz des Lernverfahrens zu garantieren. Zur Berechnung werden in der Literatur (z. B. /7/) die folgenden Gleichungen angegeben:

$$\epsilon(t) = \epsilon = \epsilon_{max} * (\epsilon_{min} / \epsilon_{max})^{t/t_{max}} \qquad \text{für } 1 > \epsilon_{max} > \epsilon_{min} > 0$$

$$\epsilon(t) = \epsilon = \epsilon_{max} * exp(-5 * (t / t_{max})^2) \qquad \text{für } 1 > \epsilon_{max} > \epsilon_{min} > 0$$

ε	: aktuelle Lernrate	t	: aktuelle Lernschrittzahl
ε_{max}	: Anfangslernrate	t_{max}	: maximale Lernschrittzahl
ε_{min}	: Untergrenze der Lernrate		

Durch die in Abhängigkeit von der Zeit (Anzahl der schon verarbeitenden Lernbeispiele) streng monoton abnehmende Funktion ε wird zunächst eine Grobstruktur erlernt, welche mit zunehmender Lernschrittzahl immer stärker verfeinert wird. Die Wahl von ε muß für jede Problemstellung getrennt erfolgen. Die Lernraten für Eingangsvektoren und Ausgangsvektoren sind in unserem Fall identisch. Es ist durchaus möglich, zwei Lernraten ε und ε' für die jeweiligen Vektoren einzuführen.

(1) Der Erregungsradius δ:

Der Erregungsradius δ legt fest, in welchem Abstand zum Erregungszentrum während des Lernvorgangs Gewichtsmodifikationen durchgeführt werden (d. h. er gibt die Reichweite der Rückkopplung an). δ ist eine Funktion der Lernschrittzahl (ähnlich wie ε). Folgende Gleichungen werden hierfür in der Literatur (z. B. /7/) angeführt:

$$\delta(t) = \delta = \delta_{max} * (\delta_{min} / \delta_{max})^{t/t_{max}} \quad \text{für } max_{r=1}^{N}\{d(r,r')\} > \delta_{max} > \delta_{min} > 0$$

$$\delta(t) = \delta = \delta_{max} * exp(-5 * (t / t_{max})^2) \quad \text{für } max_{r=1}^{N}\{d(r,r')\} > \delta_{max} > \delta_{min} > 0$$

δ	: aktueller Erregungsradius	N	: Anzahl der Neuronen
δ_{max}	: Anfangsgröße des Erregungsradius	t	: aktuelle Lernschrittzahl
δ_{min}	: Endgröße des Erregungsradius	t_{max}	: maximale Lernschrittzahl
d(r, r')	: Abstand des aktuellen Neurons r zum Neuron des Erregungszentrums r'.		

δ_{min} legt die maximale Nachbarschaftsreichweite am Ende des Lernvorgangs fest, d. h. am Ende sollte nur noch das Neuron im Erregungszentrum hinzulernen. Kleine Startwerte von δ_{max} verkürzen zwar den Lernvorgang, es besteht jedoch die Gefahr, kein vollständig geordnetes Netz zu erhalten.

Im **Schritt 2** des Algorithmus wird aus der Menge von Lernbeispielen ein Vektor zufällig ausgewählt und an den Netzeingang gelegt.

Im **Schritt 3** des Algorithmus wird das Erregungszentrum des Netzes für den in Schritt 2 am Netzeingang angelegten Vektor ermittelt. D. h. es werden die Koordinaten desjenigen Neurons ermittelt, dessen Eingangsgewichtsvektor am besten mit dem Netzeingangsvektor übereinstimmt.

In **Schritt 4 und 5** werden die Gewichtsänderungen der Ein- und Ausgangsgewichtsvektoren berechnet. Da die Lernschritte für Eingabe und Ausgabe nach dem gleichen Prinzip verlaufen werden hier nur diejenigen für die Eingangsgewichtsvektoren erläutert.

Die Gewichte für jedes einzelne Neuron r werden bei jedem Lernschritt gemäß der folgenden Lernregel geändert:

$$w_{r,in}(t+1) = w_{r,in}(t) + \varepsilon * h(r,s,\delta) * (E - w_{r,in}(t))$$

$w_{r,in}(t+1)$	: Gewichtsvektor des Neurons r nach dem Lernschritt
$w_{r,in}(t)$	: Gewichtsvektor des Neurons r vor dem Lernschritt
ε	: Lernrate (siehe oben)

E : Netzeingangsvektor

h(r,s,δ) : Erregungsfunktion (siehe unten)

Die eigentliche Gewichtsänderung wird durch das Produkt aus der Lernrate, der Erregungsfunktion und der Differenz zwischen Netzeingangsvektor und Eingangsgewichtsvektor bestimmt. Der letzte Faktor gibt dabei die Güte des bereits erlerntern Gewichtsvektors des entsprechenden Neurons an.

Der wichtigste Faktor in der Lernregel ist die Erregungsfunktion, denn durch sie fließt die Nachbarschaftsbeziehung in das Lernverfahren ein. Die Funktion lautet folgendermaßen:

$$h(r,s,\delta) = exp(-d(r,s)^2 / \delta^2)$$

r : aktuelles Neuron
s : Neuron des Erregungszentrums
δ : Erregungsradius

Die Erregungsfunktion legt fest, wie stark das Neuron r am Lernvorgang beteiligt werden soll. Dies wird über den (euklidischen) Abstand von r zum Erregungszentrum s berechnet ($d(r,s)^2$). Die Funktion nimmt ihr Maximum dann an, wenn das aktuelle Neuron gleich dem Neuron des Erregungszentrum ist. Je größer der Abstand, desto kleiner wird der Wert von h. Der euklidische Abstand wird durch die folgende Funktion ermittelt:

$$d(r,s) = d(x_r,x_s) = d((x_1,y_1),(x_2,y_2)) = \sqrt{(x_1-x_2)^2 + (y_1-y_2)^2}$$

x_s : Lagevektor des Neurons s (Erregungszentrum), aus R^2

x_r : Lagevektor des aktuellen Neurons r, aus R^2

Durch δ^2 geht der Erregungsradius in die Erregungsfunktion ein, d. h. je kleiner der Erregungsradius desto kleiner ist der Wert von h und desto weniger Neuronen im Umkreis des Erregungszentrums werden am Lernvorgang beteiligt.

3 Neuronale Netze in der Müllverbrennung

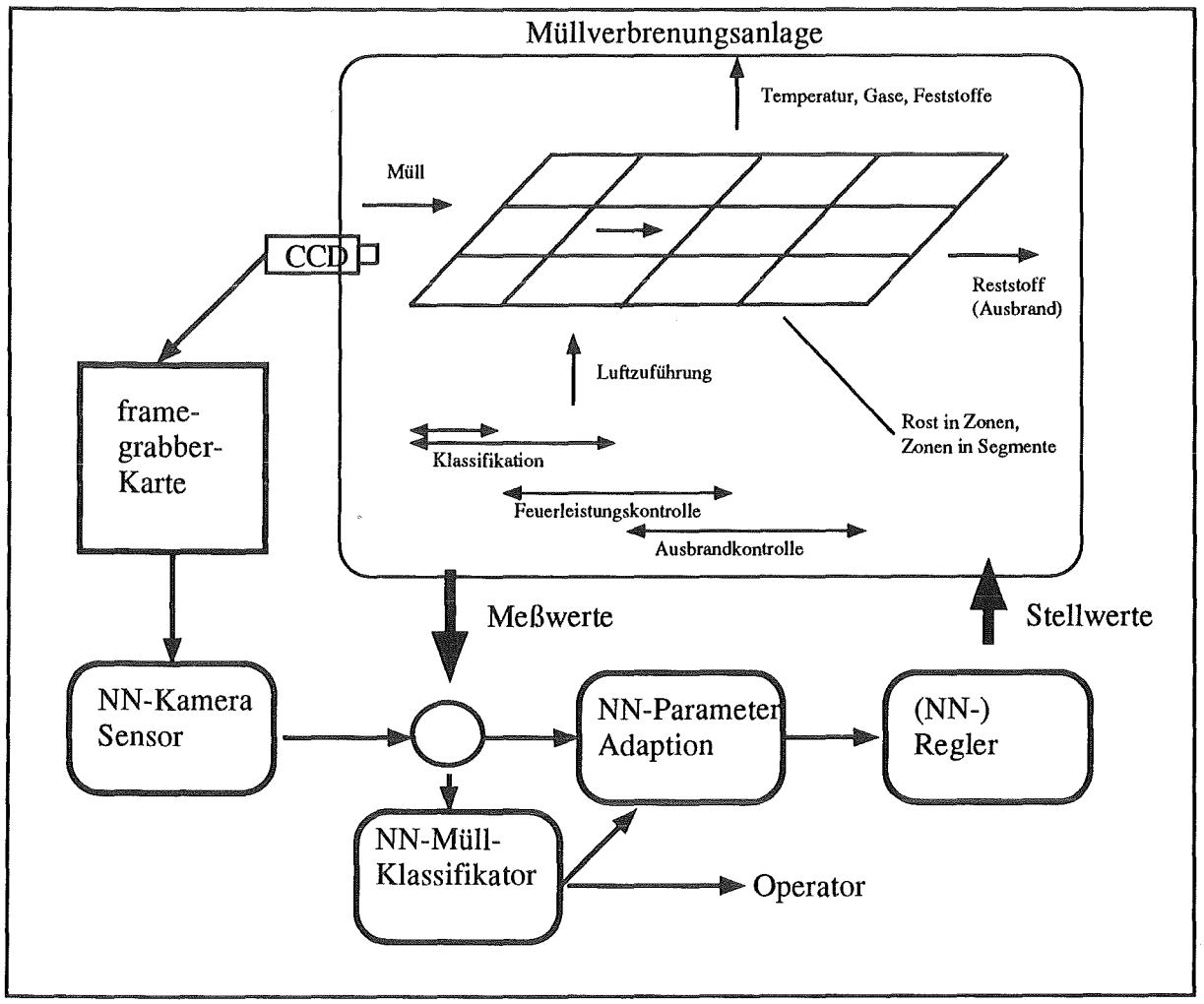
3.1 Konzept

Die Anwendungsmöglichkeiten neuronaler Netze erstrecken sich im Rahmen einer Müllverbrennung auf folgende Bereiche:

- Bilder einer Kamera des Verbrennungsprozesses werden vorverarbeitet und verdichtete Informationen (auch segmentsweise) nachfolgenden Einheiten zur Verfügung gestellt.

- Ein neuronales Netz (NN) versucht mit Hilfe ausgewählter, verdichteter Kamerawerte und weiteren Meßwerten aus dem Prozeß eine Einteilung des Mülls in Klassen (Feuchtegrad, Brennwert) nicht-überwacht zu lernen. Im laufenden Betrieb soll die entsprechend aktuelle Zuordnung vorgenommen werden. Das Ergebnis kann ein Eigenschafts-Vektor oder nur die Klassenzugehörigkeit sein.
  Andere Klassifikationsaufgaben können sich auf die lokale Brennleistung oder den Ausbrandgrad des Mülls beziehen.

- Auf der Basis der Meßwerte und der Müllklassifikation berechnet ein weiteres NN den aktuellen Parametersatz für die eingesetzten Regler (z. B. lokale Luftzuführung im Trocknungsbereich oder für die Ausbrandsteuerung).



- Der Regler kann konventionell oder auf der Basis eines NN realisiert sein,
- Als weitere Anwendung kann die Simulation des Prozesses zum Training von Anlagenfahrern oder zur Prognose (direkt oder indirekt) von Prozeßzuständen erfolgen.

3.2 Simulation

Wird ein Kohonen-Netz zur Simulation eines dynamischen Vorgangs eingesetzt, so benötigt es als Lernbeispiele die Zeitverläufe des zu simulierenden Systems. Ein Kohonen-Netze parkettiert die Trajektorienverläufe und ordnet entsprechende Folgezustände zu. Es können folgende Phasen unterschieden werden:

- Training,
- Test,
- Anwendung.

Hierbei sind Fragen bzgl.

- der Qualitätssicherung (Konvergenz / Validität),
- der Visualisierung (Dynamik / 2D-Netz / Subbereiche) sowie
- von systemdynamischen Analysen (Attraktoren / Zyklen)

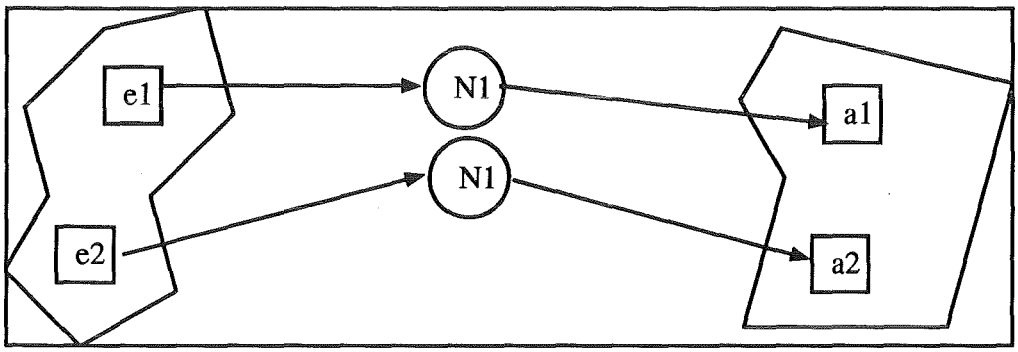
relevant. Zur Akzeptanzsicherung des NN beim Anwendermuß die Qualitätssicherung die Validität des Netzes transparent machen. In der Trainingsphase sind die Entfaltung des Netzes, die Abweichungen in den Eingangswerten der jeweiligen Neuronen und der zuzuordnenden Ausgangswerte darzustellen. Während der Testphase können Abweichungen zwischen der Netzaussage und dem realem Verlauf als Maß für die Netzgüte herangezogen werden.

3.2.1 Simulationsprinzip

Zur Nachbildung des Verhaltens eines dynamischen Systems wird einem Kohonen-Netz als Eingangsvektor die letzten vergangenen Zustände und als Ausgangsvektor der nächste zu lernende Zustand vorgegeben. Der Eingangsvektor beinhaltet auch die zum entsprechenden Zeitpunkt gültigen Stellgrößen, bzw. Systemeingangsgrößen (Luft, Rostgeschwindigkeit, Müllmasse).

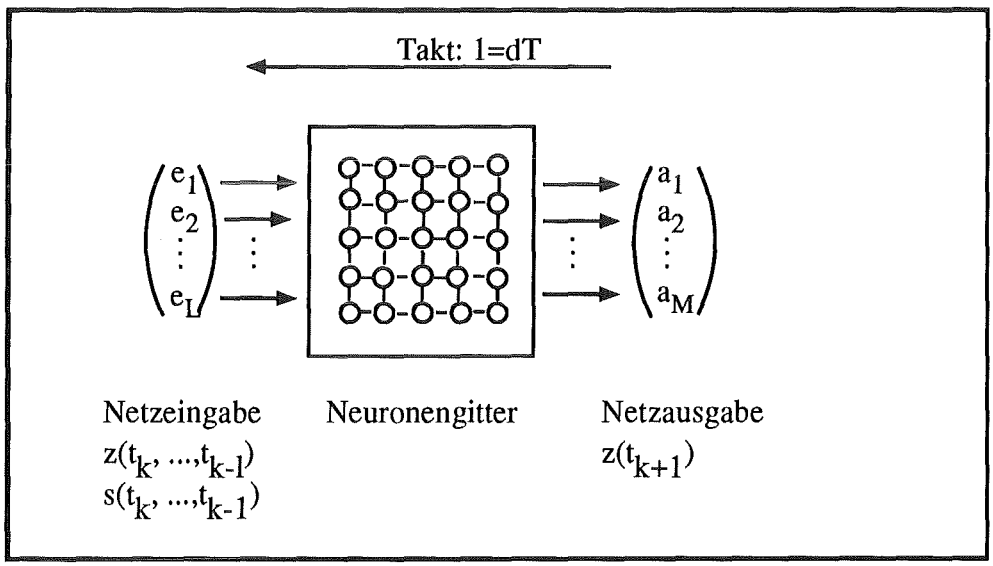
Durch seine Entfaltung unterteilt das Kohonen-Netz den Bereich der Eingangsdaten. Diese Bereichsunterteilung stellt eine Parkettierung des Weges des dynamischen Systems zum sich ergebenden Folgezustand dar. Interpretiert man die Parkettierung des Eingangsbereiches durch das Kohonen-Netze analog der numerischen Lösung einer DGl, so ergibt sich die gesuchte "Lösung" als der Ausgang des neuronalen Netzes (KN, nächster Zustandspunkt) zu einem gegebenen Eingang (Teiltrajektorie). Dabei ist jedes Neuron für einen bestimmten

Bereich zuständig. Der Eingang kann dabei aufgrund der zusätzlichen Stellgrößen eine höhere Dimension besitzen als der Ausgang.



Die Abbildung von e_i in a_i kann als f_i bezeichnet werden. Umfaßt die Menge der Ausgangsgrößen die gleichen Wertebereiche wie die identischen Eingangsgrößen, so erfolgt bzgl. diesen Größen eine Abbildung auf sich selbst. Bei einem dynamischen System ist dies als gegeben anzusehen. Sonst gäbe es Folgezustände, in denen das dynamische System verharren müßte (Anschlag).

Wird der Ausgangsvektor als neuer Teil des Eingangsvektors zurückgekoppelt und gilt $f_i = f_{i+1}$, bzw. $a_i = a_{i+1}$, so kann dieser Punkt / Bereich als Attraktionsbereich betrachtet werden, insofern er nicht am Rande des Netzes liegt. Durch eine Rückwärtsanalyse kann ein Baum mit allen Bereichen / Neuronen aufgebaut werden, die zu diesem Attraktor führen. Die zweidimensionale Darstellung ergibt somit die verschiedenen Stabilitätsbereiche, die vom Netz aus den Systemdaten gelernt wurden. Zyklen im Sinne eines oszillatorischen Verhaltens sind ebenfalls erkennbar (Pfadanalyse).



Im Gegensatz zur klasssischen Anwendung von Kohonen-Netzen darf bei der Nachbildung dynamischer Systeme keine Normierung der Eingangsvektoren erfolgen. Eine Normierung bildet alle Systemzustandsübergänge mit gleichem Winkel aber unteschiedlicher Länge des Vektors auf den gleichen Punkt auf den Einheitskreis ab; dies führt zu Fehlern.

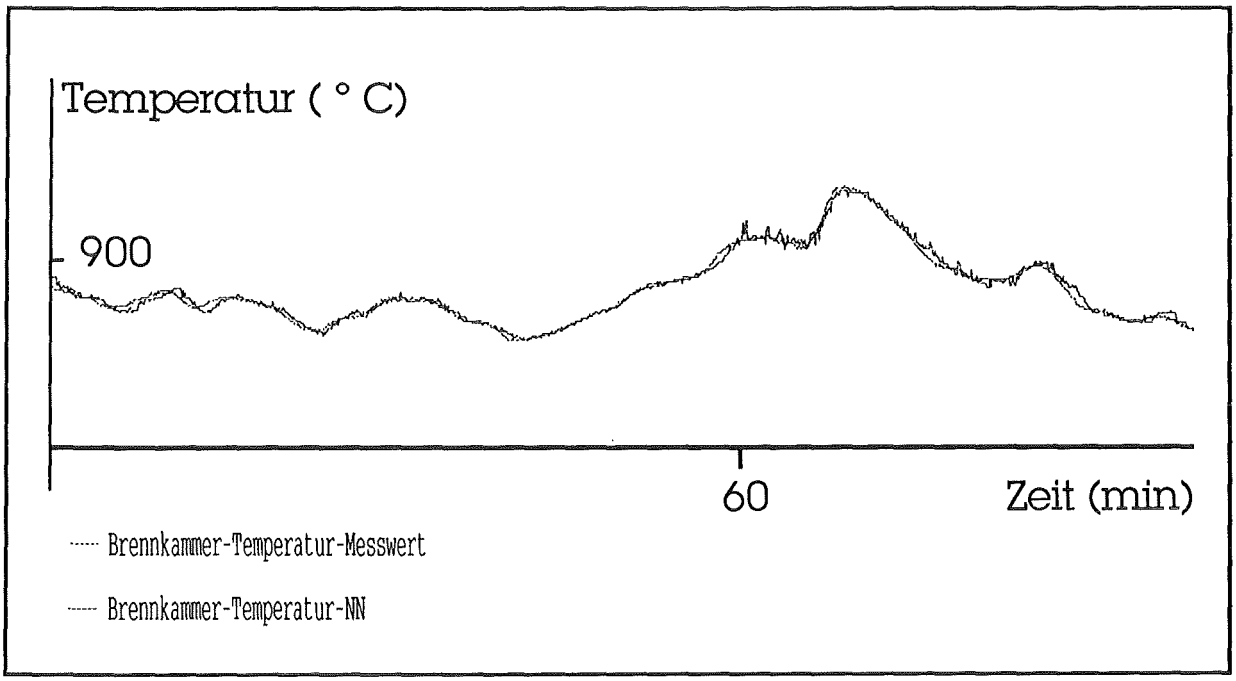
Zur Simulation eines dynamischen Systems über ein NN wird nach der Zeitdifferenz zwischen Ein- und Ausgangsgrößen der Ausgang auf die entsprechenden Eingänge zurückgekoppelt. Die Zeitdifferenz definiert den Takt der Rückkopplung.

3.2.2 Beispiel

Zwischen Brennkammertempertur (BKT) und CO, sowie O_2 ist ein grober differentieller Zusammenhang analog der folgenden Differenzengleichung erkennbar:

$$BKT\ (t+r) = f\ (BKT\ (t, .., t-n), CO\ (t, .., t-n), O_2\ (t, .., t-n)\ .$$

Dieser Zusammenhang ist von einem neuronalen Netz mit einer Prognosezeit von 1min (r=12*5s, n=3*5s, Takt=5s) gelernt worden. In der nachfolgenden Graphik ist der gemessene Brennkammertemperaturwert und der vom neuronalen Netz gelernte Wert gegenübergestellt. Es handelt sich hier um die Anwendung eines neuronalen Netzes zur direkten Prognose von Zustandsverläufen. Als Grundlage dienten allerdings ungefilterte Meßwerte, sodaß doch gewisse Abweichungen auftreten.

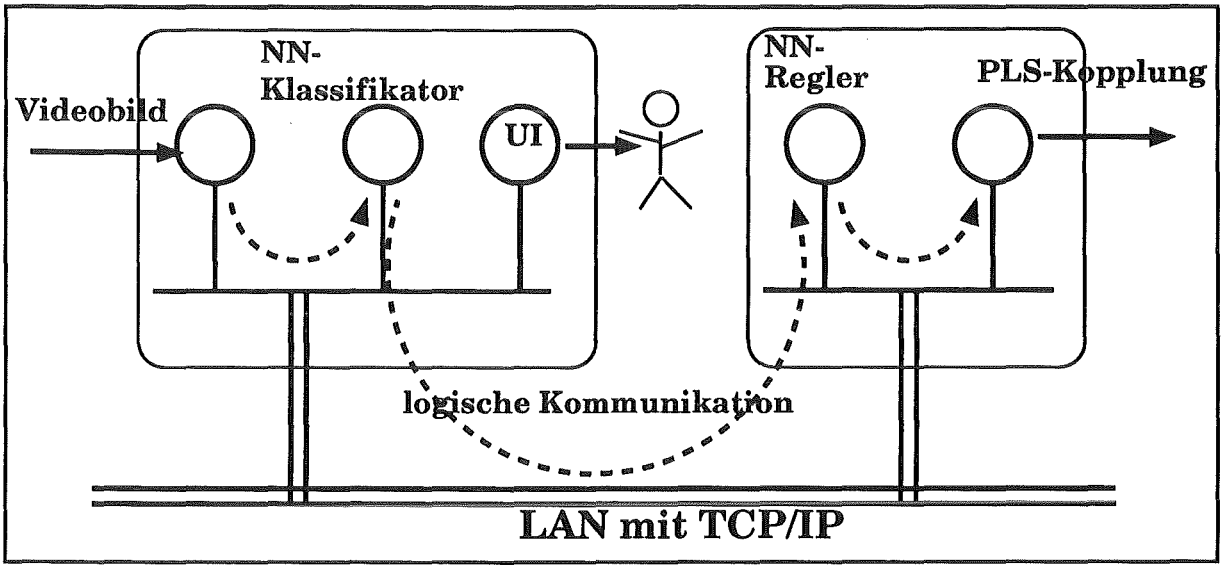


4 Das Werkzeug X-Executive

Zur Anwendung neuronaler Netze in einer Echtzeitumgebung, deren Training und Test wurde eine Entwicklungsumgebung mit Ada und X-Windows realisiert. Diese Umgebung (Experiment Executive) erlaubt die Definition und Anwendung von Experimenten für

- neuronale Netz-basierte Simulation und Prognose,
- Datenerfassung / Prozeßankopplung,
- Bildverarbeitung mit neuronalen Netzen,
- (Regelverarbeitung).

Das System besitzt eine verteilte Kommunikationsstruktur (siehe Bild), eine graphische Benutzerschnittstelle XM_View und ein Zeitreihenverwaltungssystem.



5 Zusammenfassung

Neuronale Netze können im Bereich der Umwelttechnik zur Simulation und Klassifikation komplexer dynamischer Systeme eingesetzt werden. Hierzu wurde eine Umgebung realisiert, die über eine graphische Schnittstelle den Einsatz von NN in einer Echtzeitumgebung ermöglicht (Prozeßführung von Müllverbrennungsanlagen). Zukünftige Arbeitsgebiete betreffen die systemdynamische Analyse der Netzaussagen (KN kombinieren mit linearem Assoziator), die Kombination von NN-Experiment und Regel-Experiment, die Nachbildung von NN-Fähigkeit mit symbolischen Ansätzen sowie die NN-Optimierung (Zeitverhalten). Außerdem sollen NN auf ökologische Systeme angewandt werden.

6 Literatur

/1/ Brause, Rüdiger,
Neuronale Netze,
B. G. Teubner, Stuttgart, 1991.

/2/ Keller, Hubert B.; Weinberger, Thomas,
Heuristische Modelle - ein Arbeiten mit Hypothesen?,
Proceedings des Workshops "Modellierung und Simulation im Umweltbereich",
Rostock, 25. & 26. Juni 1992.

/3/ Keller, Hubert B.; Weinberger, Thomas,
Lernmodelle und Wissensverarbeitung,
KfK-Bericht Nr. 5002, Kernforschungszentrum Karlsruhe GmbH, 1992.

/4/ Keller, H. B.; Weinberger, Th.; Kugele, E.; große Osterhues, B.,
Wissensbasierte Systeme - Darstellungs- und Verarbeitungsmodelle,
KfK-Bericht Nr. 5066, Kernforschungszentrum Karlsruhe GmbH, 1992.

/5/ Köhle, Monika,
Neuronale Netze,
Springer-Verlag, Wien, 1990.

/6/ Kohonen, Teuvo,
Self-organization and associative memory,
Springer-Verlag, Heidelberg, 3. Auflage, 1989.

/7/ Ritter, Helge; Martinetz, Thomas; Schulten, Klaus,
Neuronale Netze,
Addison-Wesley, New York, 2. Auflage, 1991.

/8/ Rojas, Raúl,
Theorie der neuronalen Netze - Eine systematische Einführung,
Springer-Verlag, Berlin, 1993

ECOBAS
Dokumentation mathematischer Beschreibungen ökologischer Prozesse

Joachim Benz *

1 Motivation, Ausgangssituation

In den vergangenen Jahrzehnten wurde weltweit in erheblichem Umfang versucht, ökologische Prozesse mit Hilfe von mathematischen Modellen zu beschreiben. Die entwickelten Modellansätze stellen eine umfangreiche Ansammlung von Wissen über die Eigenschaften und Strukturen dieser Prozesse dar. Zudem gewinnt die Anwendung der mathematischen Simulation in Analyse und Planung zunehmend an Bedeutung.
Dieses Wissen kann aber nur dann genutzt werden, wenn es

- vollständig dokumentiert und
- mit vertretbarem Aufwand verfügbar ist.

Darüber hinaus ist es wünschenswert, daß unterschiedliche mathematische Beschreibungen ein und desselben Prozesses einem wertenden Vergleich zugänglich sind. Diese Anforderungen verlangen eine vollständige Dokumentation, die

- die Beschreibung der verwendeten Gleichungen,
- die Beschreibung der Gültigkeitsgrenzen und Randbedingungen sowie
- die Beschreibung der Güte innerhalb der Gültigkeitsgrenzen

umfaßt.
Die gegenwärtige Situation entspricht nur in wenigen Fällen diesen Anforderungen. Die einzelnen Modellansätze unterscheiden sich z.T. bezüglich ihrer Frage- bzw. Aufgabenstellung, in den Zeit- und Raumskalen, dem Detaillierungsgrad sowie ihren Gültigkeitsgrenzen. Diese Unterschiede und/oder Einschränkungen der Aussagefähigkeit sind dem Außenstehenden nur in wenigen Fällen offensichtlich. Vielmehr ist meistens ein genaueres Studium der Modellansätze sowie der entsprechenden Einbettung der Entwicklungsarbeiten notwendig, um

*Universität - Gesamthochschule Kassel, FB Landwirtschaft, Internationale Agrarentwicklung und Ökologische Umweltsicherung, Abt. Futterbau und Grünlandökologie, Nordbahnhofstr. 1a, 37213 Witzenhausen, e-mail: benz@wiz.uni-kassel.de

sich diese Information zu beschaffen. Darüberhinaus fehlen in der Dokumentation der mathematischen Beschreibungen vielfach Angaben zu den Gültigkeitsgrenzen, den wichtigsten Annahmen und den Eigenschaften bezüglich der Prozessumgebung. Aufgrund der Unterschiede in den Modellansätzen ist die unmittelbare Vergleichbarkeit nur in einzelnen Fällen oder nur eingeschränkt möglich. Die Publikation der Modellansätze bzw. ihrer Anwendungen erfolgt überwiegend in Zeitschriften unterschiedlichster Fachrichtungen. Dies bedeutet, daß das oben angesprochene Wissen in heterogener Weise, verteilt und zum Teil nur unvollständig vorliegt. Die Folge dieser eingeschränkten Verfügbarkeit ist, daß die Kenntnisse z. Zt. nicht optimal und umfassend genutzt werden können [1].
Ziel des Forschungsvorhabens ECOBAS ist es, die Verfügbarkeit dieses Wissens zu verbessern. Es wird eine Datenbank aufgebaut, in der Dokumentationen mathematischer Beschreibungen ökologischer Prozesse gespeichert und abrufbar sind. Insbesondere wird auch dem Aspekt Rechnung getragen, dem späteren Benutzer bei der Auswahl für ihn geeigneter Beschreibungen möglichst weitreichend Unterstützung zu bieten. Im einzelnen werden folgende Ziele bzw. Ansprüche zugrunde gelegt.

- **leichte Verfügbarkeit der Information**
- **vollständige und eindeutige Dokumentation der mathematischen Formulierung und der Gültigkeitsgrenzen**
- **Berücksichtigung der vernetzten Struktur ökologischer Prozesse**
- **Dokumentation durchgeführter Validierungen**
- **Spezifikation der Modellgüte und von Systemeigenschaften innerhalb des Gültigkeitsbereiches**

2 Entwicklung eines Dokumentationsschemas

2.1 Dokumentationsformular

Im Rahmen des Entwurfs eines Formulars zur Dokumentation von Modellen wurde zunächst der Umfang und die Art der abzufragenden Information festgelegt, um den oben aufgeführten Zielen gerecht zu werden. Hierbei wurde auch versucht neben Erfassung der einzelnen Informationen, die Abhängigkeiten zwischen diesen Informationen zu berücksichtigen. Die Entwicklung des Dokumentationsformulars wurde in Zusammenarbeit mit

- dem Projekt UFIS (GSF, München),
- dem Ökosystemforschungszentrum Kiel und
- einigen Modellentwicklern.

durchgeführt. Das UFIS/ECOBAS-Modell-Dokumentationschema gliedert sich in folgende Hauptabschnitte:

- Allgemeine Modellangaben

- Mathematik/Algorithmik des Modells
- Technische Modellangaben
- Beschreibung der Umweltbedingungen
- Literaturreferenzen und
- Ergänzungen

Das Dokumentationsformular sowie eine Anleitung zur Bearbeitung ist unter anonymous-ftp am ftp.hrz.uni-kassel.de-Server im Verzeichnis pub/ecosys/forms oder von

Joachim Benz
Universität Gesamthochschule Kassel
FB 11
Nordbahnhofstr.1a
37213 Witzenhausen
email: benz@wiz.uni-kassel.de

Michael Knorrenschild
Forschungszentrum für Umwelt und Gesundheit (GSF)
PUC
Ingolstädter Landstr. 1
85758 Oberschleißheim
email: knorren@gsf.de

erhältlich.

2.2 Unterstützung der Dokumentation mit Hilfe eine PC-Programms

Um die Durchführung der Dokumentation zu erleichtern wird derzeit ein PC-Programm entwickelt. Neben der Erleichterung der Dokumentationsarbeit wird mit diesem Programm angestrebt, Konsitenzprüfungen des Modells durchzuführen. Diese kann dem Modellentwickler, der die Dokumentation seines Modells durchführt, wertvolle Unterstützung für seine Arbeit geben. Dieses Programm wird voraussichtlich ab dem 2. Quartal 1994 verfügbar sein.

3 Kooperation UFIS/ECOBAS

Mit Projekt UFIS, das im Forschungszentrum für Umwelt und Gesundheit (GSF, München) durchgeführt wird und eine ähnliche Aufgabenstellung verfolgt, wurde eine Kooperation vereinbart. Auch im Rahmen des Projektes UFIS wird eine Datenbank zur Dokumentation von mathematischen Modellen aufgebaut. Die Datenbank UFIS erlaubt dem Benutzer eine modellorientierte Recherche. In der Datenbank ECOBAS kann der Benutzer hingegen prozessorientiert recherchieren. Damit werden zwei Benutzergruppen mit unterschiedlichen Anforderungen angesprochen. Ein Übergang zwischen den Datenbanken ist vorgesehen. Die Zusammenhänge zwischen den beiden Datenbanken sind in Abbildung 1 dargestellt.

4 Entwurf der Datenbankstruktur

4.1 Anforderungen

Dem Entwurf der Datenbankstruktur wurden folgende Ziele bzw Ansprüche zugrundegelegt:

- Konzept: Prozess/Realisierung

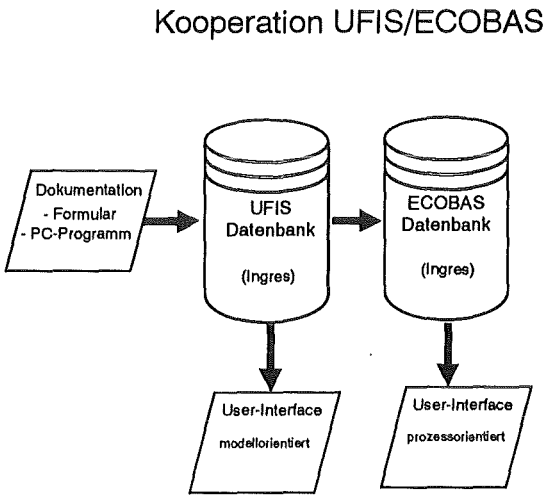


Abbildung 1: Kooperation UFIS/ECOBAS

- möglichst vollständige Dokumentation der Gültigkeitsgrenzen der Realisierungen
- Abbildung der Aggregationshierarchie in der Datenbankstruktur
- Abbildung der horizontalen Vernetzung in der Datenbankstruktur
- Erhaltung der Konsistenz innerhalb der horizontalen Vernetzung
- Speicherung der Gleichungen in einer festgelegten Syntax (TEX -Format)

4.1.1 Konzept: Prozess/Realisierung

Von zentraler Bedeutung für den Aufbau der Datenbank ist das Konzept von Prozesstypen und der Realisierung von Prozesstypen.
Wird angenommen, daß für einen bestimmten ökologischen Prozess ein oder mehrere alternative, mathematische Gleichungen als Approximationen existieren, so bedeutet es hier, es existieren ein oder mehrere Prozesstypen. Jede dieser Gleichungen hat in Abhängigkeit von der Einbettung in eine bestimmte Umgebung eine spezifische Ausprägung (Parameterwerte, Gültigkeitsgrenzen). Die Dokumentationen der spezifischen Ausprägungen sind als Realisationen eines Prozesstyps zu verstehen.

Es existiert zum Beispiel für die Primärproduktion (B) die Gleichung

$$\frac{dB}{dt} = f(B, T, L, a, b)$$

mit den Eingangsgrößen Temperatur (T) und Strahlung (L) und den beiden Parametern a und b. Bei unterschiedlichen Pflanzenbeständen oder unterschiedlichen Standorten werden a und b unterschiedliche Werte annehmen. Die Gleichung selbst entspricht dem Prozesstyp, die Dokumentation der jeweiligen Parametersätze sowie der Bezug zur

Beschreibung der jeweils zugehörigen Umwelt des konkreten Prozesses entspricht der Dokumentation der Realisierungen.

Die Dokumentation eines Prozesstyps umfaßt folgende Hauptpunkte:

- Allgemeine Angaben
  - Titel
  - Liste von Schlagwörtern
  - Kurze Beschreibung
- Deklaration
  - Eingangsgrößen
  - Ausgangsgrößen
  - Parameter
- Gleichungen
  Angabe der Gleichungen, differenziert nach PDE's, dynam. Zustandsgleichungen und algebraische Gleichungen
- Liste der eingebetteten Prozesse bzw. Funktionen
- Liste der Realisationen

Die Dokumentation der Realisationen umfaßt:

- Parameterwerte
- Verweis auf die Beschreibung der Prozessumgebung
- Gültigkeitsgrenzen
  - Eingangsgrößen
  - Zustandsgrößen
  - Zeit bzw. Zeit-Raum-Bereich
- Verweis auf die Modellbeschreibung
- Literaturverweise
- Anwendungsbereich

4.1.2 Vollständige Dokumentation der Gültigkeitsgrenzen

Um die möglichst korrekte bzw. sachgerechte Verwendung der in der Datenbank gespeicherten Informationen zu gewährleisten, ist es von besonderer Bedeutung, dem Anspruch der möglichst vollständigen Dokumentation der Gültigkeitsgrenzen der Realisationen gerecht zu werden. Siehe hierzu die Auflistung der Hauptpunkte der Dokumentation der Realisationen in Kapitel 4.1.1. Die Beschreibung der Umweltbedingungen, auf die in der Dokumentation der Realisationen nur verwiesen wird umfaßt, die Hauptpunkte:

- Allgemeine Klassifikation
  - Typ des Ökosystems (nach Ellenberg, Springstubbe)
  - geographischer Ort
  - Klimatyp (nach Walter, Lieth)
  - biologische Klassifikation
  - Bodentyp (nach FAO soil classification system)
- freie Beschreibung

4.1.3 Abbildung der horizontalen Vernetzung und der Aggregationshierarchie ökologischer Prozesse

Da in ECOBAS die Informationen nicht modellorientiert sondern prozessorientiert verwaltet und gespeichert sind, ist es notwendig die horizontale Vernetzung und die Aggregationshierarchie ökologischer Prozesse in der Datenbankstruktur abzubilden. Dadurch wird auch die Voraussetzung geschaffen, später systemanalytische Werkzeuge für die Analyse und die Weiterverarbeitung anwenden zu können.
Unter der **horizontalen Vernetzung** wird hier der folgende Sachververhalt verstanden. Bestimmte Ausgangsgrößen eines Prozesses sind Eingangsgrößen anderer Prozesse. Aufgrund dieser Zusammenhänge lassen sich Ökosysteme bzw. Ökosystemausschnitte als *Netze* von Prozessen darstellen.
Ökologische Prozesse können auf unterschiedlichen Ebenen der Aggregation formuliert werden.

Zur Verdeutlichung sei hier exemplarisch die Primärproduktion in einer stark aggregierten Form mathematisch beschrieben:

$$\frac{dBIO}{dt} = BIO * [NAR - MORT - G]$$

mit der Zustands- bzw. Output-größe BIO, der Inputgröße G (Grazing) und den eingebetteten Funktionen NAR (Nettoassimilation)sowie MORT (Mortalität bzw. Sedimentation). Zum einen existieren für diese eingebetteten Funktionen unterschiedliche mathematische Formulierungen, zum anderen lassen sie sich, je nach Grad der Aggregierung, in weitere eingebettete Funktionen aufspalten (z.B. Nettoassimilation = Photosynthese - Respiration). Dies ist in Abbildung 2 schematisch dargestellt.
Die Gesamtheit aller möglichen mathematischen Ansätze läßt sich als Hierarchie darstellen und wird hier als **Aggregationshierarchie** bezeichnet. Im Zusammenhang mit dem Konzept Prozesstyp/Realisation ergibt sich folgende Struktur, die in Abbildung 3 dargestellt ist. Die horizontale Vernetzung kann über die Dokumentation der Ein- und Ausgangsgrößen ermittelt werden.

4.1.4 Erhaltung der Konsistenz innerhalb der horizontalen Vernetzung

Durch die oben beschriebene Art der Speicherung der Information zu den ökologischen Prozessen und der Möglichkeit der Vernetzung von Prozessen kann

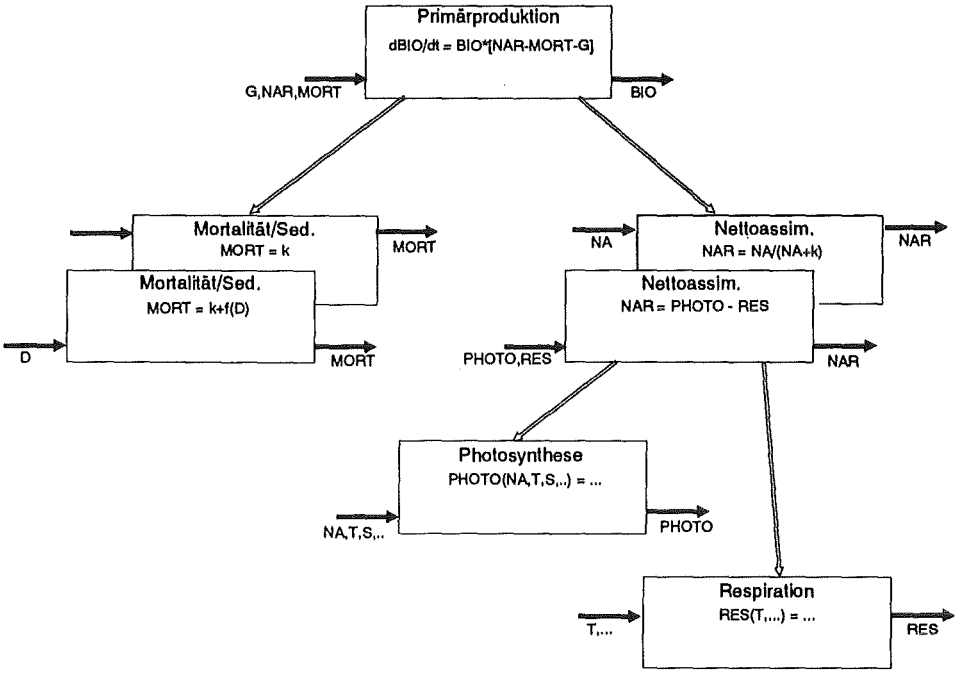


Abbildung 2: Auschnitt aus der Aggregationshierarchie

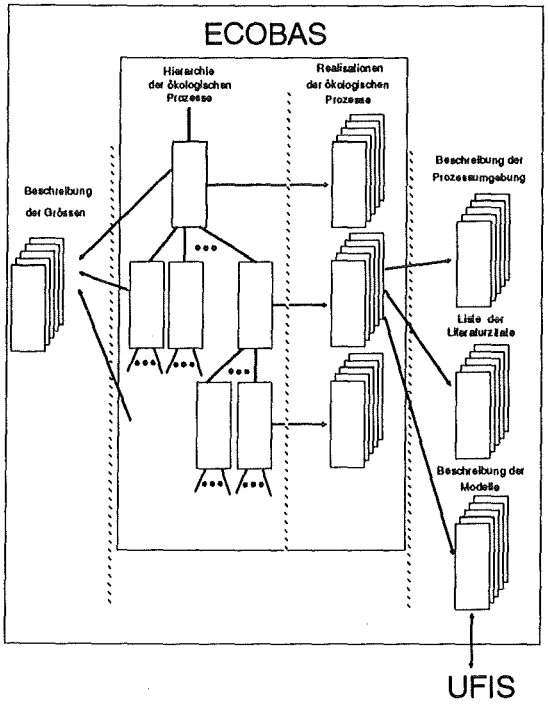


Abbildung 3: Struktur der Informationen in ECOBAS

sich im Einzelfall eine hohe Zahl von alternativen *Netzen* ergeben, die jedoch nicht alle sinnvoll bzw. zulässig sind. Es ist deshalb notwendig dem Benutzer die Möglichkeit zu geben die Konsistenz eines konkreten *Netzes* zu prüfen. Zum jetzigen Stand der Entwicklung von ECOBAS besteht die Möglichkeit über

- die Modellbeschreibung,
- die, den Realisationen zugeordnete Umweltbeschreibungen und
- die Dokumentation der Gültigkeitsgrenzen

die Konsistenz zu prüfen. Diese derzeit zur Verfügung stehenden Möglichkeiten bieten zwar wichtige und grundlegende Ansätze für die geforderte Konsistenzprüfung, eine hinreichende Überprüfung ist damit jedoch noch nicht gewährleistet. In der weiteren Entwicklungsarbeit muß zum einen geprüft werden, inwieweit zu dieser Problematik witergehende Möglichkeiten zur Verfügung gestellt werden können. Zum anderen ist geplant Schnittstellen zu schaffen, um derartige Modellhypothesen (*Netze*) möglichst einfach in andere Systeme (z.B. Simulationssysteme) zu überführen und dort zu analysieren oder zu prüfen.

4.1.5 Speicherung der Gleichungen in formalisierter Form

Für jedes Datenbankprojekt ist es wichtig, Möglichkeiten zu schaffen, daß die gespeicherten Informationen optimal genutzt werden können. Ein wesentlicher Aspekt in diesem Zusammenhang ist Nutzung der gespeicherten mathematischen Formulierungen in externen Simulationssystemen (siehe hierzu auch Kapitel 4.1.4).
Um hierzu die Voraussetzungen zu schaffen, werden die Gleichungen in der TEX-Notation gespeichert.[1] Darüberhinaus sind aber noch weitere Festlegungen z.B. für abschnittsweise definierte Funktionen oder Gleichungen, die als Tabellen vorliegen, notwendig (siehe hierzu u.a. [2]).

4.2 Datenbankstruktur

Die oben beschriebenen Anforderungen und Zusammenhänge zwischen den in ECOBAS gespeicherten Informationen ergeben die in Abbildung 4 schematisch dargestellte Datenbankstruktur. Die Datenbank ist als relationales Konzept realisiert.

5 Entwicklung der User-Interfaces

Zunächst sind in ECOBAS nur elementare Abfragetypen implementiert:

- Suche nach Formulierungen zu einem bestimmten ökologischen Prozess
- Suche nach Realisierungen eines bestimmten ökologischen Prozesses, für den bestimmte Randbedingungen zutreffen

[1] Durch die TEXNotation wird auch eine gut lesbare Form der Ausgabe ermöglicht

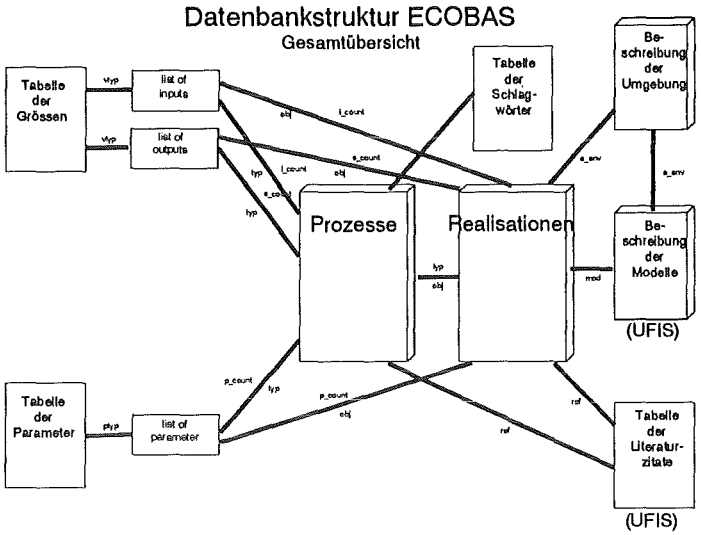


Abbildung 4: Übersicht zur Datenbankstruktur

- Suche nach untergeordneten Prozessen
- Suche nach dem übergeordneten Prozess
- Suche nach den Prozessen, bei denen eine bestimmte Größe Eingangsgröße ist
- Suche nach den Prozessen, bei denen eine bestimmte Größe Zustandsgröße ist

Für die weitere Entwicklungsarbeit ist geplant, komplexere Abfragetypen zu implementieren. Ferner ist geplant Recherchemöglichkeiten zu schaffen, die die Bearbeitung systemanalytischer Fragestellungen erlauben.

Literatur

[1] Benz,J.; M. Knorrenschild: Anforderungen an die Dokumentation mathematischer Beschreibungen ökologischer Prozesse.
in: Keller, H.B.; R. Grützner (Hrsg.): 2. Treffen des AK 5 "Werkzeuge für Simulation und Modellbildung in Umweltanwendungen". 5.11-6.11.92 in Karlsruhe. Berichte des Kernforschungszentrums, KfK 5159, 1993.

[2] Eschenbacher P.: Entwurf und Implementierung einer formalen Sprache zur Beschreibung dynamischer Systeme (Dissertation). Arbeitsberichte des Instituts für mathematische Maschinen und Datenverarbeitung, Band 23, Nummer 1. Erlangen, 1990.

Simulationsumgebungen für den Umweltbereich

Rolf Grützner
Universität Rostock
Fachbereich Informatik
AG Modellierung/Simulation
18051 Rostock

Die Simulation stellt eine der wichtigsten Problemlösungsmethoden dar, um im Umweltbereich Fragestellungen zu lösen und Entscheidungen zu treffen. Spezifische Eigenschaften der Systeme in diesem Bereich verlangen spezielle Methoden und Softwarewerkzeuge, die in ihrer Gesamtheit als Simulationsumgebung bezeichnet werden. Ihre Struktur und Gestaltung auf Grund der Anforderungen des Umweltbereiches stehen im Mittelpunkt der Darstellungen.

Hauptanwendungsbereiche der Simulation im Umweltbereich sind nach (ANGE91): Schadstoffausbreitung, wasserwirtschaftliche Untersuchungen, Steuerung technischer Prozesse und Ökosysteme. Ökosysteme stehen als Anwendungsgebiet hier im Vordergrund.

Der Schutz der Umwelt verlangt künftig die Steuerung von anthropogenen Einwirkungen, um bestimmte Grenzen der Veränderungen in der Umwelt nicht zu überschreiten (z.B. die natürliche Pufferkapazität). Zwischen beiden Systemkomponenten - dem Ökosystem und der Humansphäre - sind Rückkopplungen zu beachten. Bedingt auch durch die lückenhaften Kenntnisse über die Systeme (GRPH93), sind neuartige Modelle und Simulationsansätze notwendig. Dazu werden Lösungsvorschläge unterbreitet, die im Rahmen der Untersuchungen im SAME-Projekt erarbeitet wurden.

1. Definitionen, Begriffe

Eine Simulationsumgebung wird durch die Menge der Methoden zur Modell- und Experimentbildung, zur Experimentdurchführung, zur Ergebnisauswertung und -repräsentation sowie die Art und Weise ihrer Nutzung gebildet. Sie gliedert sich in die Modell- und Experimenterstellungsumgebung, die Experimentier-, Auswertungs- und Repräsentationsumgebung. Die Hauptmethode der Simulationsumgebung ist das Simulationslaufzeitsystem auch Simulator genannt.

Die Gestaltung der Simulationsumgebung wird wesentlich vom Einsatzbereich und den Kenntnissen der Benutzer auf dem Gebiet Informatik und Simulationstechnik beeinflußt. Andererseits beeinflußt die Simulationsumgebung selbst wesentlich die Struktur des Simulators.

Die Untersuchung dieser Wechselwirkungen, der strukturellen Konsequenzen und der Architekturen von Simulationsumgebungen und Simulatoren werden vorgestellt.

1.2. Anforderungen an Simulationsumgebungen für den Umweltbereich

Hier werden nur solche Anforderungen betrachtet, die über diejenigen an Simulationsumgebungen aus anderen Fachgebieten hinausgehen, also Anforderungen, die typisch für den Umweltbereich sind. Sie ergeben sich einerseits aus dem Fachgebiet und andererseits aus der Benutzerkategorie. Anforderungen durch die Benutzer resultieren in der Forderung nach maximaler Benutzerfreundlichkeit und Direktheit der Benutzungsoberfläche. Direktheit betrifft die Beziehung zwischen der Vorstellung des Benutzers über die Bearbeitung einer Aufgabenstellung und den Möglichkeiten, die die Oberfläche bieten. Der Grad der Direktheit einer Oberfläche ist umgekehrt proportional zum kognitiven Aufwand, den der Benutzer bei

der Bearbeitung einer Aufgabenstellung aufwenden muß, HUTC85. Die Anforderungen aus dem Umweltbereich verlangen nach neuen Methoden (z.B. zur Modellbildung, Experimentdurchführung u.a.) und Strukturen, um die anstehenden Probleme zu lösen (GRÜT93b).

Die wesentlichen Systemeigenschaften, die Modellbildung und Simulation im Umweltbereich beeinflussen, sind :

- Strukturvariabilität der Umweltsysteme
- hoher Grad unklarer Wirkungszusammenhänge (ill-defined systems)
- heterogene Systemmodelle
- sehr große Zahl von Zustandsgrößen, Parametern und Inputwerten
- die Steifheit der Systeme.

Einzelheiten sind in (KEGR93), (GRÜT93a) und (GRÜT92) publiziert. Sie ergeben sich im wesentlichen aus der Systemkomplexität und daraus, daß viele Fragen in der Ökologie und im Umweltschutz noch Gegenstand intensiver Forschung sind und oft Wirkungszusammenhänge unbekannt sind. In diesen Fällen ist eine umfassende Modellbeschreibung nicht möglich, das Modell kann nur unvollständig beschrieben werden (ill-defined). Für eine Reihe von Entscheidungen im Umweltbereich, sind aber auch Experimente mit solchen Modellen erforderlich. Zur Aufklärung der Wirkungszusammenhänge ist vorallem eine interdisziplinäre Forschung mit Biologen, Ökologen und anderen Wissenschaftlern notwendig.

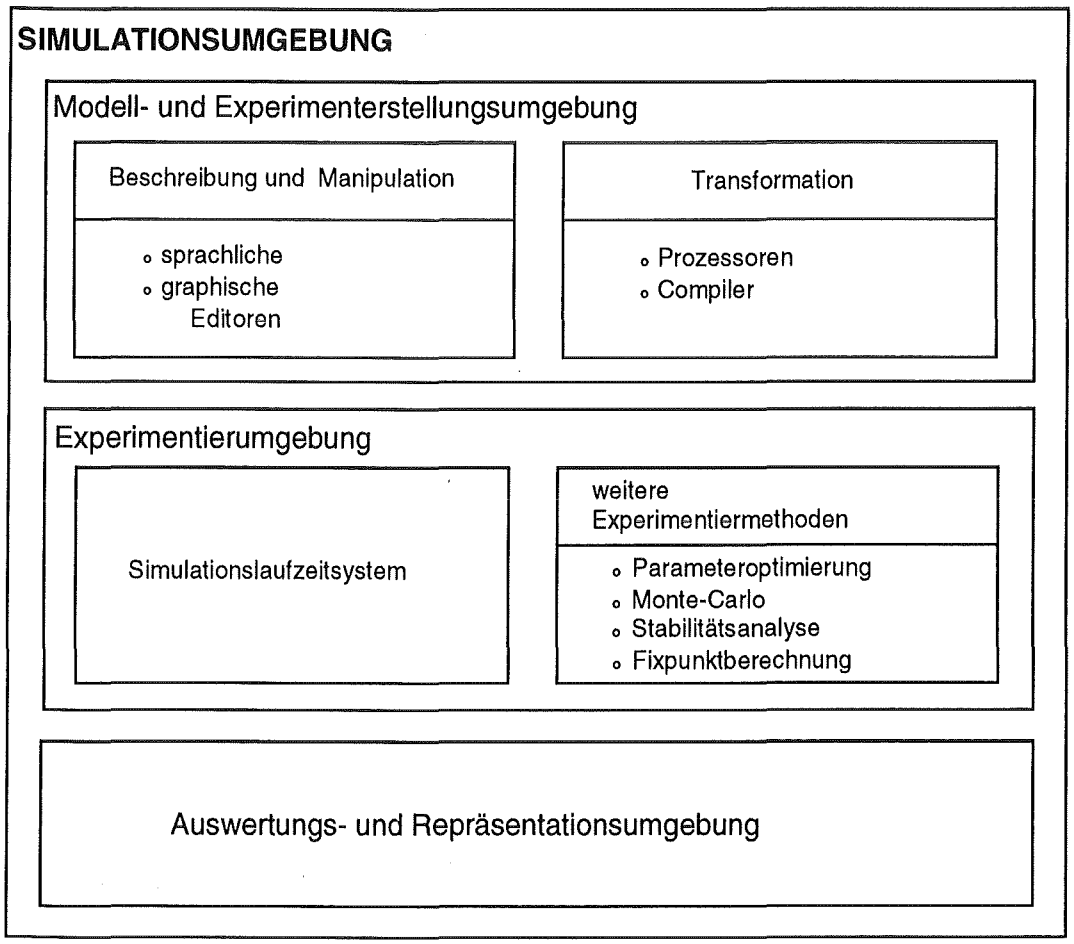


Abb.: 1 struktureller Aufbau einer Simulationsumgebung

1.3 Simulationsumgebungen

Entsprechend den obigen Angaben ergibt sich für eine Simulationsumgebung eine Struktur nach Abbildung 1.

1.3.1 Modell- und Experimenterstellungsumgebung

Modelle aus dem Umweltbereich sind modular hierarchisch aufzubauen, GRÜT93b, GRHP93, KNIJ81. Ihre Struktur ist variabel, damit erfolgt die Modellierung der Strukturvariabilität.

Ein Modell kann in Teilmodelle zerlegt werden. Teilmodelle können ihrerseits wieder aus Teilmodellen bestehen u.s.f. Teilmodelle, die nicht mehr zerlegt werden können, sind atomar, sie heißen Basisobjekte. Die Basisobjekte beschreiben die Systemdynamik. Teilmodelle heißen auch Submodelle. Auf Grund des unterschiedlichen Erkenntnisstandes über Struktur und Funktionsweise von den verschiedensten Umweltsystemen, werden die Modellbeschreibungen mit den unterschiedlichsten Werkzeugen, die dem jeweiligen Wissenstand entsprechen, vorgenommen. Das führt zu multivariabel beschriebenen Systemmodellen.

Die einzelnen Beschreibungsmittel sind: Differentialgleichungssysteme, Differenzengleichungen, prozedurale-, regelbasierte Beschreibungen, neuronale und höhere Petri-Netze sowie verschiedene automatentheoretische Konzepte in Abhängigkeit vom Anwendungsgebiet bzw. benutzereigene Beschreibungen, KNIJ81 und benutzereigener Programmcode.

Modelle, die sich aus Basismodellen zusammensetzen, die mit unterschiedlichen Beschreibungsmitteln beschrieben wurden, heißen heterogene Modelle. Solche Modelle betrachten MING90 für biotechnologische und technische Systeme, FISH91 für physikalische (Erhitzung von Flüssigkeiten), KASP93 für agrarökologische Systeme, MEKE90 für Steuerungen. Diese Autoren benutzen meist nur zwei verschiedene Beschreibungsformen. Der vorgestellte Modellieransatz durch SAME-MDL (DIMI93) ist universeller - jedes Basisobjekt kann durch einen Beschreibungstyp auf das verwendete Modellierwerkzeug hinweisen. Das Schlüsselwort "code" ermöglicht das Einfügen beliebigen Programmcodes zur Modellbeschreibung und damit die Nutzung von Fremdmodellen. Das ist für den Umweltbereich besonders notwendig, um viele bereits existierende Modelle und Programme zu nutzen. Das spart erheblichen Modellieraufwand.

Die Vielzahl der verfügbaren komplexen Modellen (u.a. auch für ein einziges ökologisches Objekt) und die Integration von Fremdmodellen stellt größte Ansprüche an die Selbstdarstellung der Modelle. Selbstdarstellung des Modells heißt, daß sich auf Grund einer geeignet formulierten formalen Modelldokumentation wesentliche funktionelle, strukturelle und inhaltliche Modelleigenschaften dem Benutzer gegenüber offenbaren. Mit Hilfe von Methoden muß aus der formalen Modelldokumentation die Selbstdarstellung des Modells (interaktiv gestützt) generiert werden. Eine solche selbstdarstellung ist notwendig in den Einsatzbereichen:

- Modellauswahl und -bildung sowie
- Modell- und Experimentnutzung.

Für beide Bereiche könnten die Ergebnisse des Forschungsprojektes ECOBAS eine mögliche theoretische Basis darstellen (BENZ92).

Von größter Bedeutung in Umweltanwendungen ist die Benutzungsoberfläche. Sie muß den Kentnissen der Benutzer Rechnung tragen. Vertiefte Informatik- und Simulationskenntnisse können bei Benutzern im Umweltbereich nicht immer vorausgesetzt werden. Das muß durch eine adäquate Benutzungsoberflächer kompensiert werden. Hier empfiehlt sich folgendes Angebot:

- eine graphische
- eine fachsprachliche: Sprache SAME-MDL
- eine programmiersprachliche (z.B. C++).

Alle drei Formen können in einem Modell gemischt werden, allerdings muß die Beschreibung der Dynamik eines Basisobjektes einheitlich erfolgen. Damit hat der Benutzer die Möglichkeit, die ihm genehme Form auszuwählen. Grundlage dieser Entscheidung war eine Biologenbefragung.

Damit der Benutzer auch ohne Syntaxkenntnisse von SAME-MDL eine Modellbeschreibung für ein Basisobjekt vornehmen kann, wird für kontinuierliche Systeme mit Zustandsraumaufspaltung (Basis: Differentialgleichungen) eine schablonenorientierte interaktive Modellschreibung eingeführt. Dabei sind nur die ohnehin notwendigen Kenntnisse über das System und allgemeine Kenntnisse zur Notation mathematischer Ausdrücke erforderlich, SAME93, GRÜT92.

In der Phase der Modellbildung beginnen die ersten Ansätze einer Modellvalidierung. Das kann unterstützt werden, indem:

- Dimensionen eingeführt werden und ihre Konsistenz überprüft wird
- jedes Basisobjekt während oder sofort nach der Modellierung -ohne zusätzliche Transformation bzw.Compilation - in einer Probesimulation überprüft wird, PLAN93.

Damit wird eine Lösung vorgeschlagen, die noch während des Modelliervorganges gestattet, Teilmodelle simulativ zu überprüfen. Diese Überprüfung kann bis zur Validierung erweitert werden. Der Vorteil des Konzeptes besteht in der schnellen Verfügbarkeit, der einfachen interaktiven Benutzung und eines frühzeitigen partiellen Korrektheitsnachweises. Der von PLAN93 in SAME realisierte Ansatz bedeutet eine Effektivitätserhöhung.

Eine Ergänzung dieser Ansätze ist durch die Angabe von Wertebereichen systemrelevanter Modellgrößen im Modell und ihre dynamische Überprüfung möglich. Jedoch führt die dynamische Überprüfung zu einer Laufzeitverlängerung.

1.3.2 Experimentierumgebung

Die Experimentierumgebung umfaßt die Art und Weise der Definition, der Beschreibung und Durchführung der Simulationsexperimente sowie die zugehörigen Methoden. Die Existenz einer Experimentierumgebung ist die unmittelbare Folge der Trennung von Modell und Experiment. Der Ansatz ermöglicht die flexible Mehrfachnutzung von Modellen, den Aufbau von Modellbanken und damit von Wissenspeichern (von Modellwissen). Er ist die Voraussetzung für Szenarienbeschreibungen und letztendlich für verbale natürlichsprachliche Experimentbeschreibungen.

Die Trennung von Modell und Experiment ist bisher nur bei sehr wenigen Systemen realisiert (SIMPLEX-II; RAMSES). Der Benutzer ist aber gezwungen, eine weitere Sprache zur Experimentbeschreibung zu erlernen. Zur Entlastung des Benutzers besteht auf diesem Gebiet Forschungsbedarf.

Der Nachteil des Ansatzes ist die Notwendigkeit einer (eigenen) Sprache zur Experimentbeschreibung. Sie dient dazu die Experimentierziele (z.B. Output- oder Parameteroptimierung, Sensivitätsanalyse) zu beschreiben, dem Modell Parameter und Inputwerte zuzuordnen und Steuerinformationen (Integrationsmethoden) anzugeben. Als Experimentbeschreibungssprache kann eine eigene Sprache aber ebenso eine höhere Programmiersprache genutzt werden. Der zweite Weg hat Vorteile. Denkbar sind auch die Zusammenfassung einer Experimentbeschreibung zu einem Experimentierkommando, das interaktiv aufgerufen die Experimentausführung startet.

Die Komplexität der Experimente im Umweltbereich erfordern die Durchführung von parallelen Simulationsexperimenten bzw. Teilexperimenten. Untersuchungen von

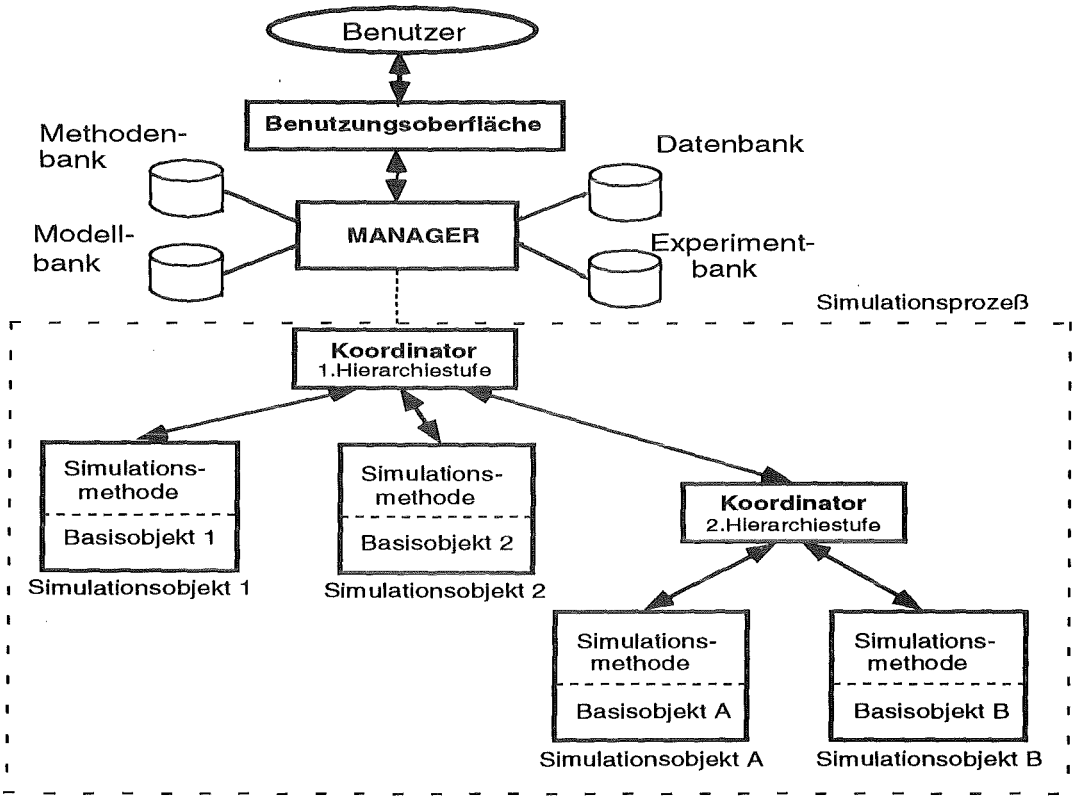


Abb.: 2 Simulatorstruktur für ein Modell nach Abb. 3

PAPE93 im Rahmen des SAME-Projektes bei der Parameteroptimierung verdeutlichen die Notwendigkeit von Parallelverarbeitung. In dem angegebenen Beispiel könnte theoretisch eine Leistungssteigerung um das 6-fache erzielt werden.

Die Beschreibung paralleler Experimente erfolgt durch das Frame-Konzept (s.u. und SAME93). Die Struktur der Simulationsumgebung wird durch parallele Experimente umfassend beeinflußt.

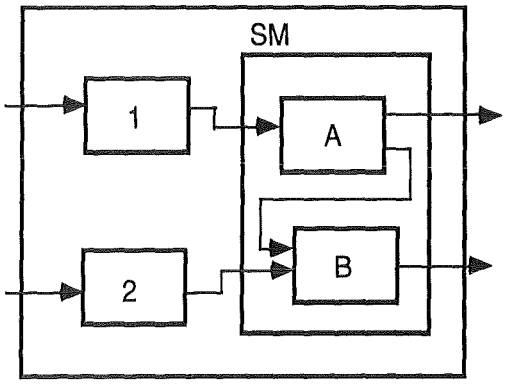


Abb.: 3 Beispiel: hierarchisches Modell

Die Experimentdurchführung (d.i. die Ausführung der Experimentbeschreibung) ist durch den Benutzer interaktiv unterbrechbar. Damit entsteht eine Eingriffsmöglichkeit in den aktuellen Rechenmodellzustand. Die Werte von Zustandsgrößen, Parametern, Zwischengrößen u.a. können ausgegeben und verändert werden. Da das Denkmodell des Benutzers das Modell ist, muß es bei derartigen Interaktionen in der Quellform auf der Benutzungsoberfläche erscheinen. In dieser Darstellung agiert der Benutzer und erhält seine Ergebnisse.

Die Hauptkomponente der Experimentierumgebung ist der Simulator. Für die Simulation heterogener modular-hierarchischer Modelle ergibt sich eine Struktur nach Abb. 2. Diese hier dargestellte Struktur entspricht einem Modell nach Abb. 3 mit zwei Basisobjekten (1 und 2) auf der 1. Hierarchiestufe und einem Submodell SM. SM enthält zwei Basisobjek-

te A und B.

Bei diesem Konzept bleibt die Modellstruktur im Simulator erhalten - die Hierarchie wird nicht "flach" gemacht. Das Konzept bietet damit die Möglichkeit der modellabhängigen Parallelität - das eine oder andere Basisobjekt wird auf dem Prozessor ausgeführt, der die erforderlichen Ressourcen (Daten, Simulationsmethode) bereitstellt. Als Konsequenz dieses Ansatzes wird eine Steuerung der Simulationsobjekte (s. Abb. 2 der Koordinator) notwendig. Sie überwacht den zeitlichen Ablauf, die statischen und dynamischen Koppelrelationen sowie das Input-/Outputverhalten auf der jeweiligen Hierarchiestufe.

Das Submodell SM in Abb.3 möge ein kontinuierliches System beschreiben; Basis sei ein System gewöhnlicher Differentialgleichungen

$$x' = f(x, s, t)$$

mit
- x Zustandsvektor
- s externer Inputvektor
- t Zeit.

Durch die Art der vorgeschlagenen Architektur wird das Gleichungssystem in ν Subsysteme zerlegt, so daß gilt:

$$x'_i = f_i(x_i, u_i, s_i, t) \qquad (1)$$

mit
$$u_i = C_i\, y = \sum_{j=1}^{\nu} C_{ij}\, y_j \qquad (2)$$

dem Kopplungsvektor.
Die Komponenten der Vektoren y werden durch den internen Subsystemoutput gebildet, der bei einem Subsystem i als Input auftritt, $y = (y_1, \ldots, y_\nu)$, wobei die Outputfunktion

$$y_i = g_i(x_i) \qquad (3)$$

den Outputvektor des Subsystems i bestimmt.

Nach Zerlegung in Teilsysteme müssen die Gleichungen (1), (2) und (3) gelöst werden. Dazu existieren iterative Ansätze und Modifikationen der bekannten Integrationsverfahren, FRAN78. Dieser Ansatz ist auf heterogene Systeme erweiterbar, allerdings sind geeignete Methoden und Fragen der Effektivität (Laufzeitverhalten) noch zu untersuchen.

1.3.3. Auswertungs- und Repräsentationsumgebung

Die Auswertungs- und Repräsentationsumgebung besitzt im Umweltbereich eine hohe Bedeutung. Dennoch werde hier nur auf einige Softwareprodukte verwiesen, die im Rahmen der Untersuchungen entstanden sind

.Neben einem Bausteinsystem für die unterschiedlichste 2D-graphische Darstellung von Simulationsergebnissen erfolgten Untersuchungen und Implemen-tationen zur vergleichenden Darstellung von Volumendaten (z.B. Schadstoffwolken) einschließlich der Bereitstellung von Funktionen zur komplexen interaktiven Manipulation an den Datenwolken STRE93a, STRE93b.

Die Visualisierung von Experimentierergebnissen verlangt oft die Datenanpassung an die Repräsentationswerkzeuge. Dazu wurde eine Datenbeschreibungssprache entworfen, die Nutzerdaten beschreibt und über einen Transformator in die für graphische Darstellung erforderliche Form überführt (STRE93b).

Literatur

ANGE91 Angerer,G.; Hiessel,H.: Umweltschutz durch Mikroelektronik:Anwendungen, Chancen, Forschungs- und Entwicklungsbedarf. VDE-Verlag, Berlin-Offenbach, 1991

DIMI93 Dimitrov, E.: Modell- und Experimentbeschreibungen für Umweltproblemstellungen. Beitrag in diesem KfK-Bericht. 1993

FISH91 Fishwick, P.A.: Heterogenes Decomposition and Inter-Level Coupling for Combined Modelling.Winter Simulation Conference Prodeedings, 1991

FRAN78 Franklin, M.,A.: Parallel Solution of Ordinary Differential Equations. IEEE Transactions on Computers, Vol. C-27, No. 5, May 1978

GRHP93 Grützner,R.; Häuslein,A.; Page,B.: Softwarewerkzeuge für die Umweltmodellierung und -simulation. in: Handbuch der Informatik, S. 129-154, Oldenbourg Verlag, erscheint 1993

GRÜT92 Grützner, R.: Simulationsumgebungen für Modell-und Experimentbeschreibungen im Umweltbereich. In: Modellierung und Simulation im Umweltbereich, Beiträge zum Workshop, Rostock, S.14-22, 1992

GRÜT93a Grützner,R.: Anforderungen an Werkzeuge zur Modellbildung und Simulation im Umweltbereich. in: KEGR93

GRÜT93b Grützner,R.: Simulationsumgebungen für die Analyse von Umweltsystemen.in: Sydow,A. (Hrsg.): Fortschritte in der Simulationstechnik.Bd. 6, Vieweg Verlag, 1993

HUTC85 Hutchins, E.L.; Hollan, J.D.; Norman,D.A.: Direct Manipulation interfaces. ICS Report 8502, Institute of Cognitive Science, University of California, San Diego, 1985

KASP93 Kaspritzki,B.: Ein Softwaresystem zur agrarökologischen Entscheidungsfindung auf Regelbasis. FB Informatik der Universität Rostock, AG Modellierung/ Simulation, 1993

KEGR93 Keller,H.B.; Grützner,R. (Hrsg.): 2. Treffen des AK 5 "Werkzeuge für Simulation und Modellbildung in Umweltanwendungen". 5.11.-6.11.1992 in Karlsruhe. Bericht des Kernforschungszentrum, KfK 5159, 1993

KNIJ81 Knijnenburg, A; Matthäus, E.; u.a.: Diskrete Simulation von ökologischen Regelmechanis men und ihre Anwendung am Beispiel eines Grobmodells der vegetativen Entwicklung von Stellaria media. in: Unger, K.; Stöcker,G. (Hrsg.): Biophysikalische Ökologie undÖkosystemforschung, Akademie Verlag Berlin, 1981

MEKE90 Meijer,R.R.; Kerkhoffs, Eugene, J.H.: Knowledge Based Process Control on a Hypercube Parallel Computer. Winter Simulation Conference Proceedings, 1990.

MING90 Ming, Rao; Tsung-Shann Jiang; Jeffrey, J.; Tsai, P.: Integrated intelligent simulation environment. Simulation, June 1990

PAPE93 Pawletta, S.; Pawletta, T.; Ewert, F.: Verifikation von Modellen agrarökologischer Systeme durch Parameteroptimierung. In Jaeschke, A. et al (Hrsg): Informatik für den Umweltschutz, 7.Symposium, Ulm , Proceedings, Springer-Verlag, Berlin, S.133-141, 1993

PLAN93 Planitzer, F.: Simulation graphisch beschriebener dynamischer Systeme. Universität Rostock, Fachbereich Informatik, AG Modellierung/Simulation,Diplomarbeit, Oktober 1993.

SAME93 Dimitrov,E.; Grützner,R.; Pawletta,S.; Pawletta,T.: SAME-Bericht.Abschlußbericht zum Forschungsprojekt SAME,abgeschlossen Dezember 1993

STRE93a Strey,K.: Darstellung räumlicher Schadstoffkonzentrationen. Universität Rostock, Fachbereich Informatik AG Modellierung/Simulation,Studienarbeit, März 1993

STRE93b Strey,K.: Interaktives Werkzeug zur Darstellung und zum Vergleich von Volumendaten. Universität Rostock, Fachbereich Informatik, AG Modellierung/Simulation,Diplomarbeit, Oktober 1993.

Modell- und Experimentbeschreibungen für Umweltproblemstellungen

Evgeni Dimitrov

AG Modellierung / Simulation
Fachbereich Informatik, Universität Rostock
A.-Einstein-Str. 21, Postfach 999, 18051 Rostock

Zusammenfassung

Die Merkmale und die Besonderheiten von Ökosystemen und die spezifischen Anforderungen aus dem Umweltbereich verlangen nach neuen Methoden und Beschreibungsmitteln. In dieser Arbeit werden Ansätze und Vorschläge für geeignete Modell-und Experimentbeschreibungsmittel vorgestellt und diskutiert. Im Mittelpunkt stehen Fragen der Beschreibung von Modellen mit Strukturvariabilität sowie der Formulierung komplexer Experimente.

Deskriptoren: Simulation im Umweltbereich, Modell- und Experimentbeschreibungen, Modelle mit Strukturvariabilität, komplexe Experimente

1. Einleitung und Motivation

1.1 Merkmale und Besonderheit von Umweltsystemen

Problemstellungen im Umweltbereich sind mit spezifischen Besonderheiten verbunden und zeichnen sich insbesondere durch eine hohe Komplexität aus. Die Menge und die Art der Relationen der Systemelemente untereinander sowie zwischen dem System und seiner Umgebung sind sehr groß. Wichtige typische Merkmale eines Ökosystems, das lebende Teilsysteme enthält, sind weiterhin (SAGE77, STGN83, NINC90):

- *dezentrale Steuerung*: Die Beziehungen zwischen den Subsystemen sind durch Ereignisbedingungen bzw. Grenzwerte charakterisiert;
- *schwache Kopplung:* Die direkte Verbindung der Subsysteme untereinander ist relativ *schwach;*
- *adaptive Veränderung*: Das synergetische Zusammenwirken mehrerer Subsysteme kann zum Überschreiten bestimmter Grenzwerte und zu sprunghaften Veränderungen des Systemzustandes führen;
- *Selbstorganisation:* Änderung der Systemstruktur im Sinne der Ein- oder Ausschaltung einiger Kopplungen zwischen den Subsystemen;
- *Selbstevolution*: Änderung des genetischen Codes der Organismen, die Systemstrukturänderung im Sinne der Generierung und Vernichtung von Subsystemen bedeutet.

Um diese Merkmale bei der Modellierung und Simulation berücksichtigen zu können sowie eine weitgehende Unterstützung der unterschiedlichen Zielstellungen im Umweltbereich zu gewährleisten wurden eine Reihe spezifischer Anforderungen an die Simulationsumgebungen und deren Komponenten aufgestellt (HÄPA88, GRÜT92, KEGR93, DIPP93).

1.2 Spezifische Anforderungen an die Modell- und Experimentbeschreibungsmitteln

Aus den Anforderungen an die Modell- und Experiementbeschreibungsmittel werden hier einige wichtige herausgegrifen und ausführlich diskutiert:

- Art und Typ der Abläufe,
- Mittel zur Darstellung spezifischer Merkmale von Ökosystemen und
- Besonderheiten bei der Beschreibung komplexer Experimente.

Die Abläufe im Umweltbereich zeigen oft einen *kombinierten* Charakter, d.h. kontinuierliche Abläufe werden fortlaufend durch diskrete Ereignisse (Diskontinuitäten) beeinflußt. Für die Darstellung solcher Abläufe kommen in erster Linie *deterministische* Modellbeschreibungen in Frage, die der Klasse der höheren kombinierten bzw. fachgebietsorientierten Simulationssprachen angehören. Die Modellbeschreibung erfolgt dann in der Regel *gleichungsorientiert* (gewöhnliche und partielle Differentielgleichungen bzw. Differenzengleichungen). Andererseits sind die ökologischen Prozesse von ihrer Natur her *stochastisch*. Oft sind die Primärdaten durch mangelhafte Zuverlässigkeit gekennzeichnet. Teilweise liegen gar keine exakten numerischen Informationen vor. In solchen Fällen werden Beschreibungen auf der Basis *stochastischer Differentialgleichungen* (TIWA78) bzw. *unscharfe* Darstellungen (fuzzy-Modellierung NERA75) bevorzugt. Methoden und Beschreibungen, die Kombinationen von stochastischen und deterministischen Prozeßanteilen berücksichtigen (BELY80), sind mit erheblichen Problemen verbunden und noch wenig erforscht (STGN83).

Zur Darstellung spezifischer Merkmale von Ökosystemen und insbesondere zur Darstellung *dynamischer Strukturänderungen* bei Selbstorganisation und Selbstevolution werden spezielle Beschreibungsmittel benötigt, die in den heutigen höheren Simulationssprachen nicht vorhanden sind. Noch problematischer ist die Beschreibung von Strukturänderungen, die durch exogene Einwirkungen hervorgerufen werden (PRAE92).

Experimentbeschreibungen spezifizieren Sachverhalte und Bedingungen, unter welchen Modelle durch Anwendung von Experimentiermethoden unterworfen und observiert werden. Somit unterscheiden sich Modell-und Experimentbeschreibungen bezüglich ihrer Charakteristika deutlich voneinander und erfordern eine *klare Trennung*. Komplexe Experimente im Umweltbereich benötigen

- unterschiedliche Experimentiermethoden,
- Beschreibungsmittel, die die Durchführung paralleler Simulationsläufe unterstützen und
- Auswahl und Aufruf mehrerer Modelle innerhalb eines Experiments bzw. Anwendung mehrerer Experimentbeschreibungen auf ein Modell.

Diese Anforderungen beeinflußen wesentlich die Modell-und Experimentbeschreibungsmittel in Simulationsumgebungen für Umweltproblemstellungen. Nachfolgend werden die Grundzüge einer höheren kombinierten Model Description Language (SAME-MDL), erweitert um Mittel zur Darstellung spezifischer Merkmale von Umweltsystemen, kurz vorgestellt. Weiterhin werden Ansätze zur Beschreibung komplexer Experimente diskutiert.

2. Grundzüge einer Modellbeschreibungssprache

2.1 Merkmale und Zielsetzungen

Beim Entwurf der Sprache wurden die Empfehlungen des CSSL-Standards (CRHA82) befolgt sowie Erweiterungen und Besonderheiten, die sich aus der Heterogenität der Modelle ergeben, berücksichtigt. Es wurde eine *deklarative zustandsorientierte Modellbeschreibung* gewählt, wie sie in modernen kombinierten Simulationssprachen (GEST, COSMOS, SIMPLEX-MDL) bevorzugt wird.

Die Modellierungstechnik orientiert sich an systemtheoretischen Konzepten. Die Beschreibung von Modellkomponenten auf unterschiedlichen Aggregationsebenen wird durch die *hierarchische Modellbildung* auf der Basis eines *Submodellkonzeptes* realisiert. Die Modularisierung ermöglicht den Aufbau und die Beschreibung *integrierter homogener Modelle.*

Die Struktur der differentialgleichungsbasierten Beschreibung von Modellkomponenten orientiert sich an dem von Ören eingeführten Layout-Konzept (ÖREN84). Bei der Wahl der Bezeichner wurde sich teilweise an die Sprache SIMPLEX-MDL(ESCH90) angelehnt. Dagegen stellen die Ansätze zur Beschreibung von Modellen mit Strukturvariabilität sowie die Integration heterogener Modellbeschreibungen *eigene Entwicklungen* dar.

Es wurde eine *vereinfachte Syntax* und *Semantik* zu Grunde gelegt, um auch unerfahrenen Nutzern eine schnelle Aneignung zu ermöglichen. Bei der Implementation der Sprache wurde auf nutzerfreundliche Techniken, wie interaktive Arbeit mit Fenstern, Menüs, Masken u.ä. zurückgegriffen.

2.2 Struktur der Modellbeschreibung

Die hierarchische modulare Modellentwicklung in SAME-MDL wird durch Verwendung von zwei Arten von Modellkomponenten,

den elementaren Bausteine, *Basisobjekte*

und

den komplexeren Bausteinen - *Submodelle*, die aus Submodellen und Basisobjekten zusammengesetzt werden können,

unterstützt

Die Basisobjekte dienen der Dynamikbeschreibung und die Submodelle der Strukturierung des Modells, wobei die Verknüpfungen zwischen den einzelnen Submodellkomponenten über Input-/Output-Ports erfolgen.
Im folgenden wird eine differentialgleichungsbasierte Beschreibung eines Basisobjektes vorgestellt. Regelbasierte Beschreibungen sind in KASP93 untersucht worden.

2.2.1 Differentialgleichungsbasierte Beschreibung eines Basisobjektes

Diese Beschreibung eines Basisobjektes (SAME93) besteht aus

- der Beschreibung des Input-/Output-Interfaces (*header description*),
- der Deklarationen der Modellelemente und deren physikalischen Maßeinheiten (*static structure*) und
- der Beschreibung der Modelldynamik, die durch algebraische Gleichungen, Differentialgleichungen und Ereignisse (*dynamic structure*) repräsentiert wird.

Das *Interface* des Basisobjektes wird durch

- den Namen der Modellklasse und
- die Deklarationen der Input-/Output-Ports (einschließlich Typangaben)

festgelegt.
Die Input-Ports vermitteln Einwirkungen aus der Systemumwelt. Die Output-Ports repräsentieren die Wirkungen des Basiobjektes nach außen.

Die Beschreibung der *statischen Struktur* enthält

- Definitionen von Maßeinheiten und deren Größenordnungen und
- Deklarationen von Modellgrößen.

Im Abschnitt "Definition von Einheiten" können neue Größen definiert bzw. vordefinierte Größen umdefiniert werden. Als vordefinierte Größen stehen die im SI-Einheitensystem aufgenommenen Maßeinheiten zur Verfügung. Bei unterschiedlichen Größenordnungen können die Umrechnungsfaktoren in einer *unit*-Deklaration angegeben werden.
Die Deklaration der Modellgrößen umfaßt Konstanten, Zustandsvariablen, Hilfsvariablen(abhängige Variablen die durch algebraische Gleichungen definiert sind) und Funktionen (Standard-, Tabellen- und nutzerdefinierte Funktionen). Zulässige Typen für die Modellgrößen sind **int, float** und **double**. Jede Modellgröße kann mit einer Maßeinheit (standard oder selbsdefiniert) versehen werden.

Die *Dynamikbeschreibung* besteht aus
- algebraischen Gleichungen,
- Differentialgleichungen und
- Ereignissen.

Die Beschreibung der algebraischen und Differentialgleichungen erfolgt in der üblichen mathematischen Notation. So wird ein Differentialquotient syntaktisch durch ein einfaches Hochkomma hinter dem Bezeichner,

<state_var_name> '= <expression>

dargestellt.

Bei Modellen mit variablem Modellverhalten kann der Zustandsraum eines Basisobjektes in *Regionen* aufgeteilt werden. Jede Region enthält dann einen unterschiedlichen Satz von Beschreibungsgleichungen.

Ereignisse beschreiben Aktionen und deren Auslösung in Abhängigkeit vom Wahrheitswert einer Ereignissbedingung. Die Ereignissbedingung kann zeitabhängig oder zustandsabhängig sein. Die Ereignissaktionen beschreiben
- Zustandsübergänge oder
- die Umschaltung zwischen Regionen.

Durch einen diskreten Zustandsübergang verändert eine kontinuierliche Zustandsvariable sprunghaft ihren Wert.
Zur Darstellung von Ereignissen dient die **when**-Anweisung:

when *(<event_condition>)*
{ <event_statements>} .

Diese Anweisung löst eine Ereignisaktion nur dann aus, wenn der Wahrheitswert der Ereignissbedingung von FALSE auf TRUE wechselt. Dabei erfolgt eine Synchronisation des Integrationsprozesses mit dem Ereigniszeitpunkt.

2.2.2 Beschreibung eines Submodells

Die *Submodell-Beschreibung* ist in die Abschnitte
- Beschreibung der Komponentenschnittstelle,
- Beschreibung der Submodellkomponenten und deren Verbindungen und
- Initialisierung von Input/Output-Ports

gegliedert.

Die *Komponentenschnittstelle* eines Submodells ist mit dem Interface eines Basisobjekts identisch.

Die *Beschreibung der Submodellkomponenten* enthält die Aufzählung der einzelnen Subkomponenten. Um heterogene Modellkomponenten in einem Submodell integrieren zu können, muß ein Identifikator der Beschreibungsart dem Modellkomponentennamen vorangestellt werden. Vorgesehene Identifikatoren sind
- **differential** - für differentialgleichungsbeschriebene Basisobjekte,
- **difference** - für differenzengleichungsbeschriebene Basisobjekte,
- **rule** - für regelbasierte Basisobjekte und
- **code** - für Beschreibungen in einer höheren Programmiersprache.

Die *Verbindungen* (Connections) repräsentiert die inhaltliche Abhängigkeit zwischen den einzelnen Teilkomponenten und verdeutlicht den Kommunikationsfluß im Modell.

Zur Darstellung von Strukturänderungen (Selbstorganisation und - evolution) werden zwei Gruppen von Sprachkonstrukten bereitgestellt. Die Konstrukte

delete_component(<component_identifier>),
create_component(<component_identifier>) und
exchange_component(<old_component_identifier , new_component_identifier>)

beschreiben Operationen über Komponenten und

delete_connection(<connection_description>)
create_connection(<connection_description>)

Operationen über Verbindungen.

Die Beschreibung von dynamischen Strukturveränderungen erfordert
die Festlegung einer Initial-Struktur und
die Ergänzung der Initial-Strukturbeschreibung um *Strukturmodifikaionsbeschreibungen*, welche in Abhängigkeit von Bedingungen (durch Ereignisse dargestellt) die Strukturveränderungen spezifizieren (Abb. 1).
Die Angabe der Initial-Struktur ist optional. Ist eine Initial-Struktur angegeben, so beziehen sich alle Änderungen, die durch **when**-Anweisungen beschrieben werden, auf die Initial-Struktur. Wenn keine Ereignissbedingung erfüllt ist, gelten die Strukturbeschreibungen der Initial-Struktur. Bei Submodellbeschreibungen ohne Initial-Struktur enthalten die **when**-Anweisungen eine unabhängige, vollständige Beschreibung einer Modellkonfiguration. Die Ereignisbedingungen müssen in diesem Fall eine vollständige Zerlegung des Ereignisraumes bewirken (keine Überschneidung von Bedingungen).

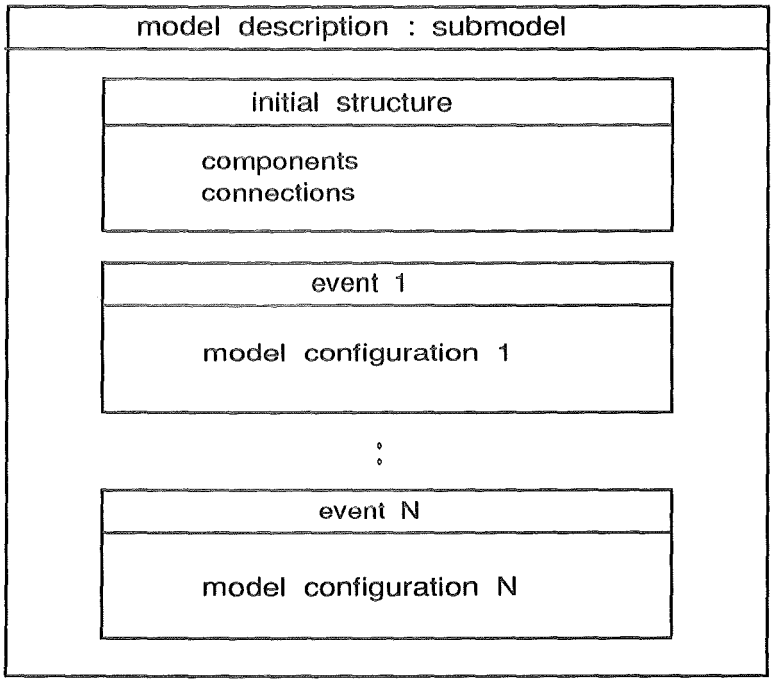


Abb. 1: Submodellbeschreibung mit Strukturvariabilität

Die Verwendung der Sprache SAME-MDL bei der Beschreibung von Submodellen mit Strukturvariabilität wird im folgenden anhand eines Beispiels aus der Agrarökologie, der Ontogenese von Winterweizen (PAPE93) demonstriert. Die Beschreibung der Ontogenese erfolgt in einzelnen Phasen, deren Anzahl und deren Einflußparameter variieren. Ein Phasenübergang erfolgt in Abhängigkeit vom Ontogenesezustand. Das Modell wird aus verschiedenen vorgefertigten Teilmodellen zusammengestellt (Abb.2).

Die Phasenübergängen werden in der Submodellbeschreibung durch Strukturänderungen dargestellt. Für jede Phase gilt eine andere Modellkonfiguration. In Abb.3 ist die Beschreibung eines zweiphasigen ArcWheat-Modells:
Phase 1: Basisobjekte O und V
und
Phase 2: Basisobjekte O, TL und LT
angegeben .

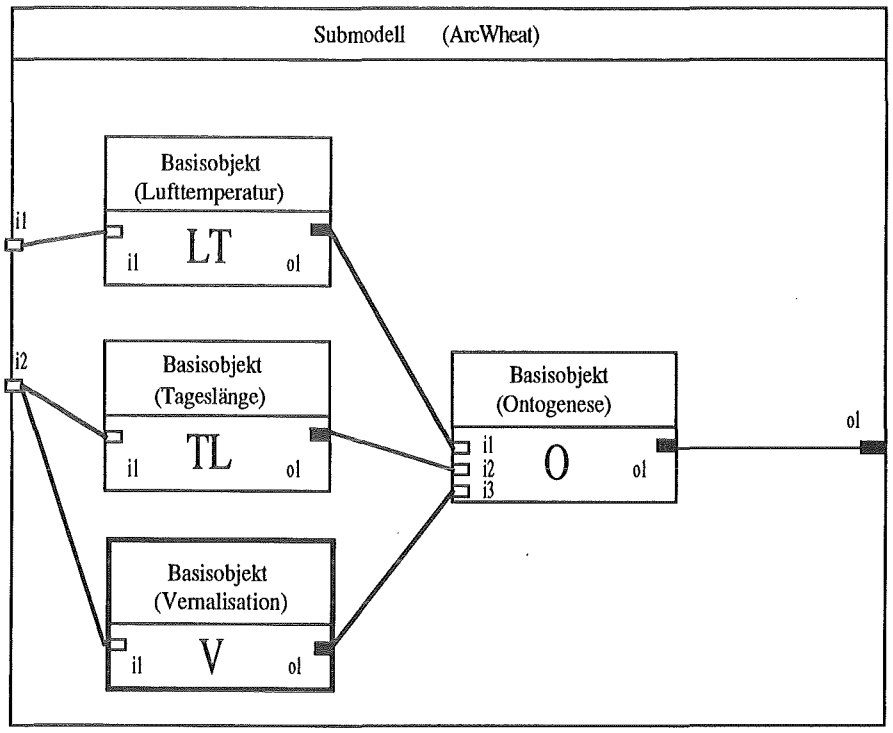


Abb. 2: Graphische Darstellung des Winterweizen-Modells

```
submodel ArcWheat(inp: double t1, double i1, int i2; out: double o1);
initial                    /* phase 1 */
  components
    O(inp: double i3; out: double o1);
    V(inp: int i1; out: double o1);
  connections
    V.o1 -> O.i3;
    i2 -> V.i1;
when (time > t1)      /* phase 2 */
{ create_component(TL(inp: int i1; out: double o1));
  create_component(LT(inp: double i1; out: double o1));
  delete_component(V);
  create_connection(i2-> TL.i1);
  create_connection(i1-> LT.i1);
  create_connection(TL.o1-> O.i2);
  create_connection(LT.o1-> O.i1);
  delete_connection(i2 -> V.i1);
  delete_connection(V.o1 -> O.i3);
}
end.   /* submodel ArcWheat */
```

Abb. 3: SAME-MDL-Beschreibung eines Submodelles mit Strukturvariabilität

3. Ansätze zur Beschreibung komplexer Experimente

3.1 Komplexe Experimente und Formen des Experimentierens

Komplexe Experimente werden durch Anwendung von unterschiedlichen Experimentiermethoden auf Modelle realisiert. Die Mehrzahl der Experimentiermethoden steht bereits in verschiedenen Softwarepaketen zur Verfügung. Solcher bereits existierenden Methoden sollten durch Integration in einer Experimentierumgebung verfügbar gemacht werden.

Man unterscheidet grundsätzlich zwei Formen des Experimentierens (WITT92),

- *interaktives* Experimentieren - Arbeit mit Kommandos einer Kommandosprache,
- *prozedurale* (*sprachliche*) Experimentierbeschreibung - Arbeit mit Anweisungen einer Beschreibungssprache, wobei auch Kommandos aus der interaktiven Umgebung verwendet werden können.

Das interaktive Experimentieren bedeutet die interaktive Anwendung von Experimentiermethoden (Initialisierung des Modells, Durchführung von Simulationsläufen, Speicherung von Experimentdaten) auf Modelle.

3.2 Sprachliche Beschreibung komplexer Experimente

Die sprachliche Experimentbeschreibung ist Grundlage

- einer automatischen (d.h. durch das System) Ausführung von Experimenten,
- der Beschreibung komplexer Experimente, wie z.B. Kopplung von Simulationsmodellen und Optimierungsmethoden, und
- einer präzisen und übersichtlichen Dokumentation der Experimente.

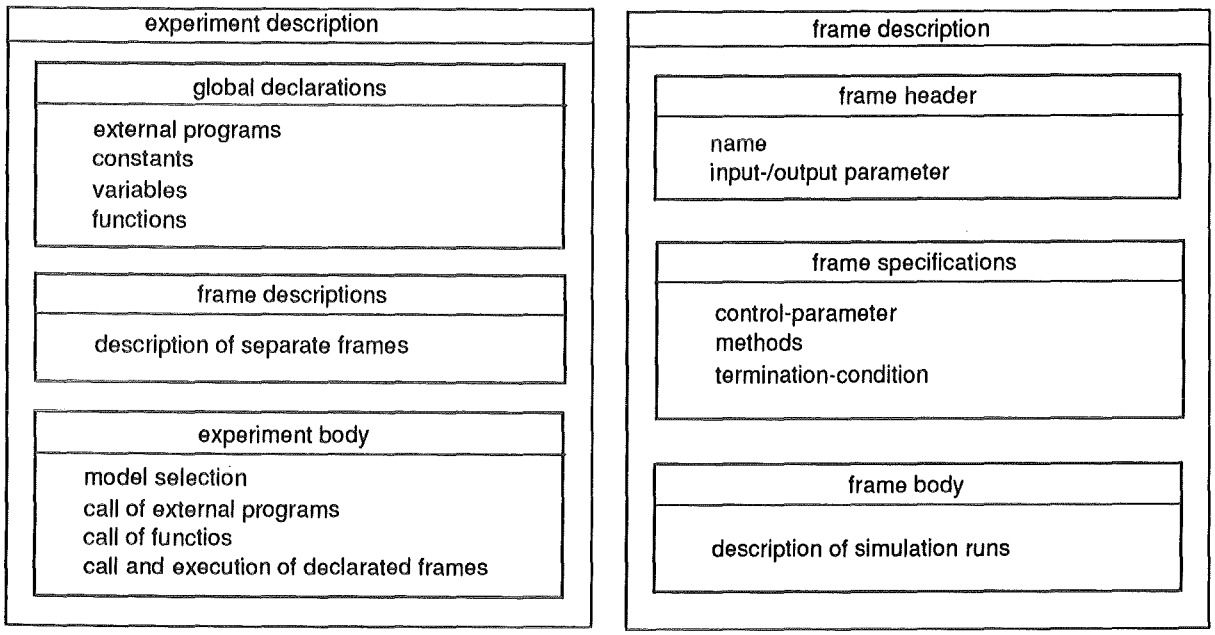
Zur sprachlichen Beschreibung komplexer Experimente nach DIPP93 sind Mittel

- zur Darstellung von Algorithmen innerhalb der Experimentbeschreibung (*Funktionen*),
- zur Beschreibung *multipler* Simulationsläufe und
- zur Strukturzerlegung des Experimentes

erforderlich.

Die Nutzung einer höheren Programmiersprache für die Experimentbeschreibung, wie z.B. C++ (ELWE93), ist zwar mit geringem Implementationsaufwand verbunden, aber eine Programmiersprache ist in der Regel umfangreicher als notwendig und nicht auf spezielle Anforderungen zugeschnitten.

Eine spezielle Experiment Description Language bietet geeignetere Strukturierungsmöglichkeiten bei der Experimentzerlegung (Abb. 4 a). Die einzelnen Struktureinheiten, hier *Frames* genannt (Abb. 4 b), enthalten neben prozeduralen Sprachanweisungen auch interaktive Kommandofolgen sowie simulationsspezifische Anweisungen. Das *Frame*-Konzept ermöglicht damit die Verschachtelung von (Teil-)Experimenten. Die Beschreibung bedingungsabhängiger wiederholbarer Abschnitte im Experimentkörper verlagert sich in die einzelnen Frames und Funktionen. Ein Frame kann auf unterschiedliche Modellbeschreibungen angewendet werden. Die Zuordnung zwischen Frame und Modell erfolgt im Körper des (Haupt-)Experimentes. Er besteht weiterhin aus Aufrufen externer Programme, lokaler Funktionen und vordeklarierter Frames.
Das Frame-Konzept bietet günstige Voraussetzungen zur Paralellisierung eines Experimentes. "Unabhängige" Frames, d.h. Frames, die keine Daten untereinander austauschen, können parallel abgearbeitet werden.



a) b)

Abb.4 a) Struktur einer sprachlichen Experimentbeschreibung
b) Struktur eines Frames

4. Ausblick

Der in dieser Arbeit vorgeschlagene Ansatz zur Beschreibung von Modellen mit dynamischen Strukturänderungen bietet gute Unterstützung bei Modellen mit "geringer" Strukturvariabilität. Bei Modellen, die dynamisch eine Reihe von unterschiedlichen Konfigurationen aufweisen, entstehen schnell lange und zum Teil unübersichtliche Beschreibungen. In solchen Fällen wird eine allgemeine (generalisierte) Strukturbeschreibung aller möglichen Modellvarianten (-konfigurationen) benötigt. Aus dieser Beschreibung können dann durch Reduktion automatisch einzelne Modellvarianten generiert werden. Als geeignete Darstellungsform für die Reduktion bieten sich regelbasierte Beschreibungen an.

5. Literatur

BELY80 Belyaev, V.J.: On the evaluation of potential productivity of marine ecosystems by mathematical modelling, ISEM Journal vol.2, 1980, pp. 8-16

CRHA82 Crosbie, R.E. , Hay, J.L.: Towards New Standards for continuous System Simulation Languages". Proc.1982 SCSC, Denver, Colorado (Juli 19-21, 1982), pp.186-190

DIPP93 Dimitrov, E., Pawletta, S., Pawletta, T.: Anforderungen an Simulationssysteme unter dem Aspekt der Durchführung komplexer Experimente (Parameteroptimierung). 6. Workshop "Simulation und künstliche Intelligenz", 22./23.04.1993, Univ. Karlsruhe In: ASIM-Mitteilungen 1993, (Heft in Druck)

ELWE93 Elwert, T.: Experimentiermethoden in der Umgebung eines kombinierten Simulators, Diplomarbeit, Universität Rostock, Fachbereich Informatik, Rostock, 1993

ESCH90 Eschenbacher, P.: Entwurf und Implementierung einer formalen Sprache zur Beschreibung dynamischer Modelle. Dissertation, Univ. Erlangen-Nürnberg, Erlangen, 1990

GRÜT92 Grützner,R.: Simulationsumgebungen für Modell- und Experimentbeschreibungen im Umweltbereich. in: Grützner,R.; Möller,A.; Pawletta,T. (Hrsg.):Beiträge zum Workshop "Modellierung und Simulation im Umweltbereich",Rostock 25.6.-26.6.92, Gesell schaft für Informatik, GI-Regionalgruppe Rostock/Wismar und Universität Rostock, FB Informatik, 1992

HÄPA88 Häuslein,A.; Page,B.: Anforderungen an interaktive Simulationssysteme für die Umweltanalyse. In: Jaeschke, A.; Page, B. (Hrsg): Informatikanwendungen im Umweltbereich, 2.Symp.,Proceedings, Informatik-Fachberichte Bd.170, Springer-Verlag, Berlin, S.102-115, 1988

KASP93 Kaspritzki,B.: Ein Softwaresystem zur agrarökologischen Entscheidungsfindung auf Regelbasis. FB Informatik der Universität Rostock, AG Modellierung/ Simulation, 1993

KEGR93 Keller, H.B.; Grützner, R. (Hrsg.): 2. Treffen des AK "Werkzeuge für Simulation und Modellbildung in Umweltanwendungen". 5.11.-6.11.1992 in Karlsruhe. Bericht des Kernforschungszentrum, KfK 5159, 1993

NERA75 Negotai, C.V.; Ralescu, D.A.: Applications of fuzzy sets to systems analysis. Birkhäuser, Basel, 1975

NINC90 Ninck, A.: Objektorientierter Ansatz zur Simulation von lebenden Systemen. In: Simulationstechnik, 6.Symposium, Tagungsband, Sept.1990, Vieweg-Braunschweig, 1990

ÖREN84 Ören, T.I.: GEST- A Modelling and Simulation Language Based on System Theoretic Concepts. In: Ören,T.I; Zeigler,B.P. and Elzas, M.S. (Eds): Simulation and Model-Based Methodologies: An Integrative View, North-Holland, Amsterdam, 1984, 281-335

PAPE93 Pawletta, S.; Pawletta, T.; Ewert, F.: Verifikation von Modellen agrarökologischer Systeme durch Parameteroptimierung. In Jaeschke, A. et al (Hrsg): Informatik für den Umweltschutz, 7.Symposium, Ulm , Proc. Springer-Verlag, Berlin, S.133-141, 1993

PRAE92 Praehofer, H: Modelling and Simulation. In Pichler, F.; Schwärtzel,H. (Eds.): CAST Methods in Modelling, Springer-Verlag, 1992, pp. 123-241

SAGE77 Sage, A.P.: Methodology for large-scale systems, McGraw-Hill, New York, 1977

SAME93 Dimitrov,E.; Grützner,R.; Pawletta,S.; Pawletta,T.: SAME-Bericht.Abschlußbericht zum Forschungsprojekt SAME, abgeschlossen Dezember 1993

STGN83 Straskraba, M.; Gnauck, A.: Aquatische Ökosysteme - Modellierung und Simulation, Akademie Verlag, Berlin, 1983

TIWA78 Tiwari, J.L. et al. : Some stochastic differential equation models of an aquatic ecosystem, Ecol. Modelling vol. 4, 1978, pp. 3-27

WITT92 Wittmann, J.: Die Durchführung von Experimenten im Simulationssystem SIMPLEX-II. Tutorium: SIMPLEX II, Ebernburg, Bad Münster a. Stein, 25.3. - 26.3.1992

ZEIG86 Zeigler, B.P.: Toward a Simulation Methodology for Variable Structure Modeling. In: Elzas,M.S.; Ören,T.I.; Zeigler, B.P.(Eds): Modeling und Simulation Methodology in the Artificial Intelligence Era, North-Holland Pub.Co., Amsterdam, 1986, 281-335

Entwicklung einer tabellenorientierten, grafischen Simulationsmethodik zur Modellierung von Verkehrsträgeremissionen

H. Freese, A. Häuslein, I. Isbarn, A. Klee, B. Page, J. Seidel
Fachbereich Informatik
Universität Hamburg

1. Einleitung

Durch das Simulationssystem Dynamis IIx werden die Mittel der grafischen Modellerstellung für die kontinuierliche Simulation bereitgestellt. Die kontinuierliche Simulation ist die Grundlage für die Modellierung einer Reihe von Problemstellungen aus dem Bereich der Ökologie. Die grafische Modellierung bietet die Möglichkeit, komplexe Zusammenhänge auf überschaubare Art und Weise darzustellen.

Grafische Modellierung setzt Modellen hinsichtlich der Anzahl der verwendeten Modellgrößen jedoch Grenzen. Ein Modelldiagramm erscheint nur dann übersichtlich, wenn nicht zu viele Verbindungen die Sicht verstellen. Ein Mittel zur Beherrschung von Komplexität ist die Aggregation. Eine ursprünglich große Zahl von Einzeldaten/ -objekten wird zu einer, sie im Modell repräsentierenden, Größe zusammengefaßt. Nachteil dieses Vorgehens ist jedoch der Verlust von Information, denn eine Rückprojektion des Simulationsergebnis ist nur sehr eingeschränkt vorzunehmen.

Für eine Reihe von Fragen wäre ein niedriger Aggregationsgrad des Simulationsergebnisses wünschenswert. Die entstehenden Daten ermöglichen differenziertere Aussagen. Um Ergebnisse auf abstrakter Ebene zu erhalten, kann eine nachträgliche Bearbeitung mit statistischen Werkzeugen erfolgen.

Ein weiteres auf dem Gebiet der kontinuierlichen Simulation bisher wenig berücksichtigtes Thema ist die Verwendung von empirischen Meßdaten. Empirische Daten gehen in ein kontinuierliches Modell vor allem bei der Beschreibung der Ausgangssituation und beim Modellentwurf ein. Ihre direkte Einbindung in den Simulationslauf ist wegen ihres Umfangs bisher kaum möglich. Die Verwendung historischer Daten als Grundlage könnte die Realitätsnähe eines Modells in Bezug auf quantitative Aussagen erhöhen. Unter Umständen würde dies die Modellvalidierung vereinfachen.

Einen Ansatz für beide Themen versucht das vorliegende Modellierungswerkzeug zu bieten.

2. Das Simulationssystem Dynamis IIx

Dynamis IIx ist ein an der Universität Hamburg entwickeltes Simulationssystem (vgl. (Häuslein, 1992) u. (Grützner 1993)). Es ist konzipiert worden, um den Aufwand der Modellerstellung und der Simulationsdurchführung zu verringern. Zu nennen sind hier vor allem die Möglichkeit der grafischen Modellierung und die wissensbasierte Unterstützung der Modellentwicklung.

2.1. Grafische Modellierung

Ein Modell wird grafisch durch ein Modelldiagramm in einem Diagrammfenster dargestellt. Ein Diagramm beinhaltet Modellgrößen in bildlicher Darstellung. Die Größen besitzen ihrer Rolle entsprechend differenzierte grafische Symbole. Sie werden mit der Maus erzeugt und

angeordnet. Zwischen den Modellgrößen werden Verbindungslinien gezogen. Hierdurch kann ein großer Anteil des Entwurfs auf grafischer Ebene stattfinden. Die Gleichungen der Modellgrößen werden durch die über die Diagrammsymbole zugänglichen Inhaltsfenster editiert. In ihnen werden numerische Werte und arithmetische Ausdrücke zur Spezifikation des Verhaltens der betreffenden Größe angegeben. Häufig ergeben sich diese Informationen bereits aus der Modellgrafik und können automatisch generiert werden. Grafische Modellierung bietet dem Benutzer einen anschaulichen, intuitiven Zugang zur Modellerstellung. Grafische Modelle sind ohne Programmieraufwand herstellbar. Ein Diagramm dient bereits in der Erstellungsphase der Modelldokumentation und bietet einen einfachen Zugang für weitere Modellnutzer.

2.2. Wissensbasierte Unterstützung

Die Wissenbasis enthält Wissen über die genutzte Simulationsmethodik. Das Wissen ist hier in Form von Regeln und Aussagen kodiert. Es wird dem Nutzer auf verschiedene Art zur Verfügung gestellt. Aktionen des Benutzers werden während des Modellentwurfs auf ihre Vereinbarkeit mit diesen Regeln überprüft. Dies betrifft die Plazierung der Modellgrößen und das Ziehen von Verbindungen. Zum Beispiel sind nicht alle Modellgrößen frei kombinierbar oder beliebig oft mit anderen Modellgrößen zu verbinden. Eine Regelverletzung, also ein Ausbrechen aus der Simulationsmethodik, führt zu einer Warnung und in einigen Fällen zu einem Abbruch der Aktion. Unter Verwendung der Erklärungskomponente der Expertensystemshell kann sich ein Modellersteller weitere Erläuterungen verfügbar machen, die den Sachverhalt näher ausführen oder Alternativen aufzeigen.

Nach Abschluß der Erstellung, bevor ein Simulationslauf durchgeführt wird, findet eine Prüfung des Gesamtmodells statt. Hier wird die Modellvollständigkeit geprüft, also ob alle für die Simulation notwendigen Größen und Verbindungen existieren. Weiterer Prüfungsgegenstand ist eine eindeutige Namensgebung innerhalb eines Teilmodells, sowie die Zyklenfreiheit der Modellverbindungen, die für die Berechnung wichtig sind.

2.3. Erweiterbarkeit

Die Verwendung objekt-orientierter Techniken für die Programmierung und die Integration einer Expertensystemshell ermöglichen eine weitgehende Unterstützung des Modellerstellungsprozesses ohne Verlust von Flexibilität. So beinhaltet das System eine Reihe verschiedener Simulationsmethodiken und bietet Beratungsmöglichkeiten für ihren Einsatz unter bestimmten thematischen Randbedingungen. Die Einführung neuer Simulationsmethodiken unter Nutzung der vorhandenen Klassenhierarchie ist im Systemkonzept vorgesehen und dadurch erleichtert worden.

Die Systemarchitektur teilt das System in drei Ebenen: die externe, die interne und die wissensbasierte Ebene. Die externe Ebene stellt die Benutzeroberfläche, also die grafische Darstellung der Modellgrößen zur Verfügung. Auf der internen Ebene sind die Funktionen zur Modellauswertung und der Simulationsdurchführung zu finden. Die wissensbasierte Ebene beinhaltet, wie schon beschrieben, Wissen über Modellaufbau und Modellkorrektheit. Die Trennung dieser drei Ebenen vereinfacht die Erweiterung des Systems. Änderungen können auf den verschiedenen Ebenen vorgenommen werden, ohne daß die anderen Ebenen davon beeinträchtigt würden.

Es ist deswegen als Plattform für prototypische Entwicklungen gut geeignet.

3. Methodik der Verkehrsemissionsmodellierung

3.1. Anforderungen der Verkehrsemissionsmodellierung

Das Erstellen von Modellen zur Berechnung von Emissionen des Pkw-Verkehrs sollte durch ein Modellierungswerkzeug erleichtert werden. Das Werkzeug muß das Einlesen gegebener Tabellen unterstützen und die Möglichkeit bieten, Tabellen in Relation zu setzen, um arithmetische Operationen auf ihnen auszuführen. Es muß zusätzlich Statistikfunktionen und Ausgabemöglichkeiten zur Verfügung stellen.

3.2. Entwurfsziel des Modellierungswerkzeugs

Die Datensammlung auf dem Gebiet der Verkehrsträgeremissionen liegt in Form von Tabellen vor. Ziel der entworfenen Simulationsmethodik *Dynamic-Relations* ist es, grafische Modellierung und die Nutzung tabellarisch dargestellter Daten zu vereinen.

Für die Verwendung von Tabellen existiert auf dem Gebiet der Datenbanken ein umfangreiches Instrumentarium. Aufgabe der Simulationsmethodik *Dynamic-Relations* ist es, weite Teile dieses Instrumentariums bereitzustellen und es um eine einfach zu verwendende Möglichkeit der Verrechnung von Spalteninhalten zu ergänzen. Über die Verwendung empirischer Meßdaten hinaus soll es Möglichkeiten geben, Hochrechnungen und Prognosen zu erzeugen. Ein Simulationslauf besteht zum einen aus der Nachvollziehung und Aufbereitung historischer Verläufe anhand empirischer Daten und zum anderen aus dem Erzeugen von Szenarien durch Prognosen und Hochrechnungen.

Die Funktionalität der Modellgrößen wurde so gewählt, daß sie die Verknüpfung von Tabellen, die Verrechnung von Wertemengen, Projektion und Selektion aus Tabellen, sowie Aggregation und Gruppierung von Werten ermöglicht.

4. Die Modellgrößen

Die Simulationsmethodik *Dynamic-Relations* umfaßt eine Reihe von Modellgrößen, die sich durch ihre Rolle in einem Modell unterscheiden. Das Verhalten einer Größe wird über ihre *Definition* beschrieben. Definitionen sind mathematische Ausdrücke, die zu Tabellen oder zu Beschreibungen der Tabellenkonstruktion evaluieren (vgl. (Freese u. Seidel, 1993)). Verbindungen im Modelldiagramm beschreiben die Werteflüsse der Größen untereinander. Innerhalb der Definition einer Modellgröße treten die Namen anderer Modellgrößen, zu denen eine Verbindung besteht, auf. Die Werteübertragung kann auch entgegen der Verbindungsrichtung verlaufen.

Im Falle der Eingangstabellen können die ihnen zugehörigen Dateien schon während des Modellentwurfs geladen werden. Die weiteren Größen erzeugen ihre Werte durch Auswertung ihrer Definition.

Berechnungstabelle	Verknüpfen von Tabellen
Eingangstabelle	Laden externer Daten
Schlüssel	Bilden eines Schlüsselausdrucks
Selektor	Selektion und Projektion
Ausgabe-Selektor	Darstellung einer Wertereihe in Kurvenform
Operationsvorschrift	Bilden eines arithmetischen Ausdrucks
Operation	Arithmetisches Verknüpfen verschiedener Spalten einer Tabelle
Aggregator	Aggregieren und Gruppieren
Submodell	Modularisierung in Teilmodelle
Inout	Kommunikation zwischen Teilmodellen

Tab. 4.1. Übersicht der Modellgrößen

Berechnungstabelle (Table)

Diese Größe verknüpft eine beliebige Anzahl Tabellen zu einer Ergebnistabelle. Der Benutzer gibt die Spalten, die in die Ergebnistabelle eingehen sollen, an. Tabellen werden strukturerhaltend oder strukturverändernd verknüpft. Strukturerhaltende Funktionen sind die Mengen-

operationen Vereinigung und Durchschnitt. Tabellen müssen für diese Operation verträglich sein, das heißt, sie müssen gleiche Spaltenanzahlen und ihre Spalten denselben Typ besitzen. Operanden der Mengenoperationen sind die Zeilenmengen der zu verknüpfenden Tabellen.

Strukturverändernde Tabellenverknüpfungen sind das Kreuzprodukt oder die Join-Operation (vgl. (Zehnder, 1987)).

Eingabetabelle (Input-table)

Eine Eingabetabelle stellt einem Modell externe Daten zur Verfügung. Zu diesem Zweck verfügt sie über zwei verschiedene Funktionen, Daten in Tabellenform zu laden. Die Funktion LOAD lädt Dateien, deren Name mit dem Namen der Modellgrößen übereinstimmen. Diese Dateien enthalten empirische Daten in textueller Form. Eine weitere Funktion PROGNOSIS lädt eine Tabelle, deren Werte nicht durch konkrete Zahlen sondern durch Funktionen beschrieben wurden. Aus ihnen werden Prognosewerte errechnet.

Schlüssel (Key)

Diese Größe bildet Schlüsselausdrücke. Ein Schlüsselausdruck assoziiert Spaltennamen und Wertemengen und beschreibt so die vom Selektor zu selektierende Zeilenmenge. Der Spaltenname bezeichnet eine Spalte in der Tabelle, auf die der Schlüssel wirken soll. Die Wertemenge beschreibt eine Teilmenge der Zeilen dieser Tabelle. Die Teilmenge entsteht, indem die Zeilen selektiert werden, die in der betreffenden Spalte zur Wertemenge passende Werte besitzen. Die Wertemenge kann offene und geschlossene Intervalle und die Wildcards ALL und ANY umfassen. ALL steht für alle Werte der betreffenden Spalte, ANY für den zuerst auftretenden passenden Wert.

Selektor (Selector)

Die Selektorgröße führt die, durch einen Schlüssel näher definierte, Selektion einer Zeilenmenge aus ihrer Quelltabelle durch. Sie nimmt zusätzlich eine Auswahl der Spalten vor, die die neue Tabelle enthalten soll.

Ausgabe-Selektor (Output-selector)

Der Ausgabe-Selektor wirkt ähnlich wie ein Selektor. Die durch ihn selektierten Wertemengen sind auf genau einen Wert pro Simulationszeitpunkt eingeschränkt. Aus dem ermittelten Wert und dem gerade erreichten Stand der Simulationszeitführung bildet diese Modellgröße eine Zeitreihe. Sie ermöglicht eine laufbegleitende Ausgabe in Form einer Kurve an einer Zeitachse.

Operationsvorschrift (Instruction)

Innerhalb dieser Größe stehen Spaltennamen als Bezeichner für die ihnen zugeordneten Wertereihen. Aus ihnen wird ein arithmetischer Ausdruck zur Verrechnung ganzer Spalteninhalte gebildet. Die Größe leitet den Ausdruck an die Operationsgröße weiter, die die Berechnung vornimmt.

Operation (Operation)

Die Operationsgröße berechnet den durch die Operationsvorschrift erzeugten Ausdruck. Sie bildet eine neue Tabelle aus einer ausgewählten Teilmenge der Spalten der Ausgangstabelle und

fügt ihr die neu berechneten Ergebnisse in Form von neuen Spalten an. Diese neuen Ergebnisspalten erhalten den Namen der sie erzeugenden Instructiongröße.

Aggregator (Aggregator)

Der Aggregator modifiziert seine Quelltabelle, indem er die Werte ausgewählter Spalten aggregiert oder gruppiert. Eine Aggregation findet durch die Funktionen Minimum, Maximum, Summe, Mittelwert und Standardabweichung statt.

Eine Tabelle wird gruppiert, indem die Zeilen, die in einer bestimmten Spalte denselben Wert besitzen, auf eine Zeile abgebildet werden. Die zusammenfallenden Werte werden durch eine Aggregatfunktion zusammengefaßt.

Submodell (Submodel)

Ein Submodell dient der Modularisierung umfangreicher Modelle. Es tritt in einem Modelldiagramm als Modellgröße auf. Ein Submodell repräsentiert jedoch seinerseits wieder ein eigenes Modelldiagramm. Verbindungen zu/von einer Submodellgröße repräsentieren die Verbindungen in und aus dem Submodell heraus. Submodelle können von verschiedenen Metamodellen in verschiedenen Kontexten genutzt werden.

Inout (Inout)

Inout-Größen dienen der Kommunikation zwischen den diagramminternen Größen eines Submodells mit seiner Umgebung. Sie befinden sich im Modellinterface.

5. Beschreibung des Emissionsmodells

Im Emissionsmodell werden mehrere Tabellen herangezogen, die gemessene oder statistische Daten des Pkw-Verkehrs enthalten (vgl (Niederle, 1991)). Mithilfe dieser Eingangs-Tabellen werden die Emissionen des Pkw-Verkehrs berechnet. Die Daten stammen von verschiedenen Institutionen (u.a. TÜV Rheinland, Umweltbundesamt, ifeu-Institut Heidelberg). Die verwendeten Tabellen beschreiben die Überlebenskurven, die Zulassungsanteile, die Neuzulassungen, die mittlere Fahrleistung, die Strassenartanteile und die Emissionsfaktoren. Diese Eingangs-Tabellen beinhalten folgende Daten:

* Überlebenskurven-Tabelle: In dieser Tabelle liegen für jedes Bezugsjahr Prozentwerte für die Überlebenswahrscheinlichkeit der Fahrzeuge jeder Altersklasse (0 bis 25 Jahre).
* Zulassungsanteil-Tabelle: Die Zulassungsanteile sind die Anteile eines Fahrzeugtypen am Gesamtbestand und beziehen sich auf ein Jahr.
* Neuzulassungs-Tabelle: Diese Tabelle enthält die Zulassungszahlen für jedes Jahr.
* Fahrleistungs-Tabelle: Hierin sind die Fahrleistungen eines Jahres enthalten, jeweils aufgeteilt nach Fahrzeugtyp und Alter.
* Strassenartanteil-Tabelle: Die Zuordnung der Anteile bezieht sich auf die Straßenarten Innerorts, Außerorts und auf Bundesautobahnen, da Strassen abhängig von Alter und Fahrzeugtyp unterschiedlich frequentiert werden.
* Emissionsfaktoren-Tabelle: Für jede Schadstoffkomponente, jeden Fahrzeugtyp und jede Straßenart liegen unterschiedliche Emissionsfaktoren vor.

Die Berechnung der Emissionen erfolgt in drei Schritten. In jedem Schritt wird eine neue Tabelle erzeugt. Die Zwischenergebnisse "mittlere Fahrleistung" und "Bestand" sollen die Übersichtlichkeit erhöhen.

Im ersten Schritt (Bestandsberechnung) werden die Tabellen "Überlebenskurven", "Zulassungsanteile" und "Neuzulassungen" zu einer weiteren Tabelle verbunden. Innerhalb dieser neu erzeugten Tabelle können dann die entsprechenden Operationen ausgeführt werden, um die Bestandszahlen zu produzieren. Im zweiten Schritt werden Werte der Tabellen "mittlere Fahrleistung", "Strassenartanteile" und "Bestand_nach_Alter" zur Berechnung der Tabelle "Fahrleistung_je_Typ_nach_Strassenart_und_Alter" herangezogen. Diese wird wiederum mit den "Emissionsfaktoren" verknüpft, um die gewünschten "Emissionen" zu erhalten. In dieser Tabelle liegen dann die Resultate der Emissionsberechnung vor. Sie sind unterteilt nach Straßenart, Fahrzeugtyp und Schadstoffkomponente.

Die Ergebnisse können mit verschiedenen Funktionen aufbereitet werden. Z.B. können alle Schadstoffemissionen eines Jahres summiert oder in einer Grafik abgebildet werden.

6. Zusammenfassung und Ausblick

Ein prototypische Realisierung der hier vorgestellten Simulationsmethodik *Dynamic-Relations* ist zur Zeit in Arbeit. Erfahrungen bei der Implementation der methodischen Erweiterung und bei der Benutzung des Prototyps zur Modellerstellung werden Rückwirkungen auf die Konzeption haben. Insbesondere der Umgang mit Prognosen und Hochrechnungen bedarf noch weiterer konzeptioneller Arbeit. Eine weitere Perspektive ist die Integration von Daten in Tabellenform in die Simulationsmethodik SYSTEM-DYNAMICS (vgl. (Forrester, 1968)). Insbesondere die dieser Methodik zugrundeliegende Betonung von Regelkreisen und Rückkopplungen wären eine interessante Ergänzung.

Die hier kurz vorgestellte Simulationsmethodik *Dynamic-Relations* ist für die speziellen Anforderungen der Simulation von Verkehrsträgeremissionen entwickelt worden. Das von ihr erfaßte Problemgebiet ist jedoch nicht durch diese Anwendung begrenzt.

Literatur:

Forrester, J.W. (1968)
Principles of Systems. Wright-Allen Press, Cambridge

Freese, H., Seidel, J. (1993),
Objekt-orientierte Konzeption und Implementation von Methoden eines wissensbasierten Simulationssystems.
Eingereicht als Studienarbeit am Fachbereich Informatik Universität Hamburg

Häuslein, A. (1992)
Wissensbasierte Unterstützung der Modellbildung und Simulation im Umweltbereich, Konzeption und prototypische Realisierung eines Simulationssystems.
Peter Lang Verlag, Frankfurt

Häuslein, A. (1993)
Konzeption eines wissensbasierten Simulationssystems zur Unterstützung der Modellbildung und Simulation im Umweltbereich.
In: (Jaeschke, 1993), S. 199-210

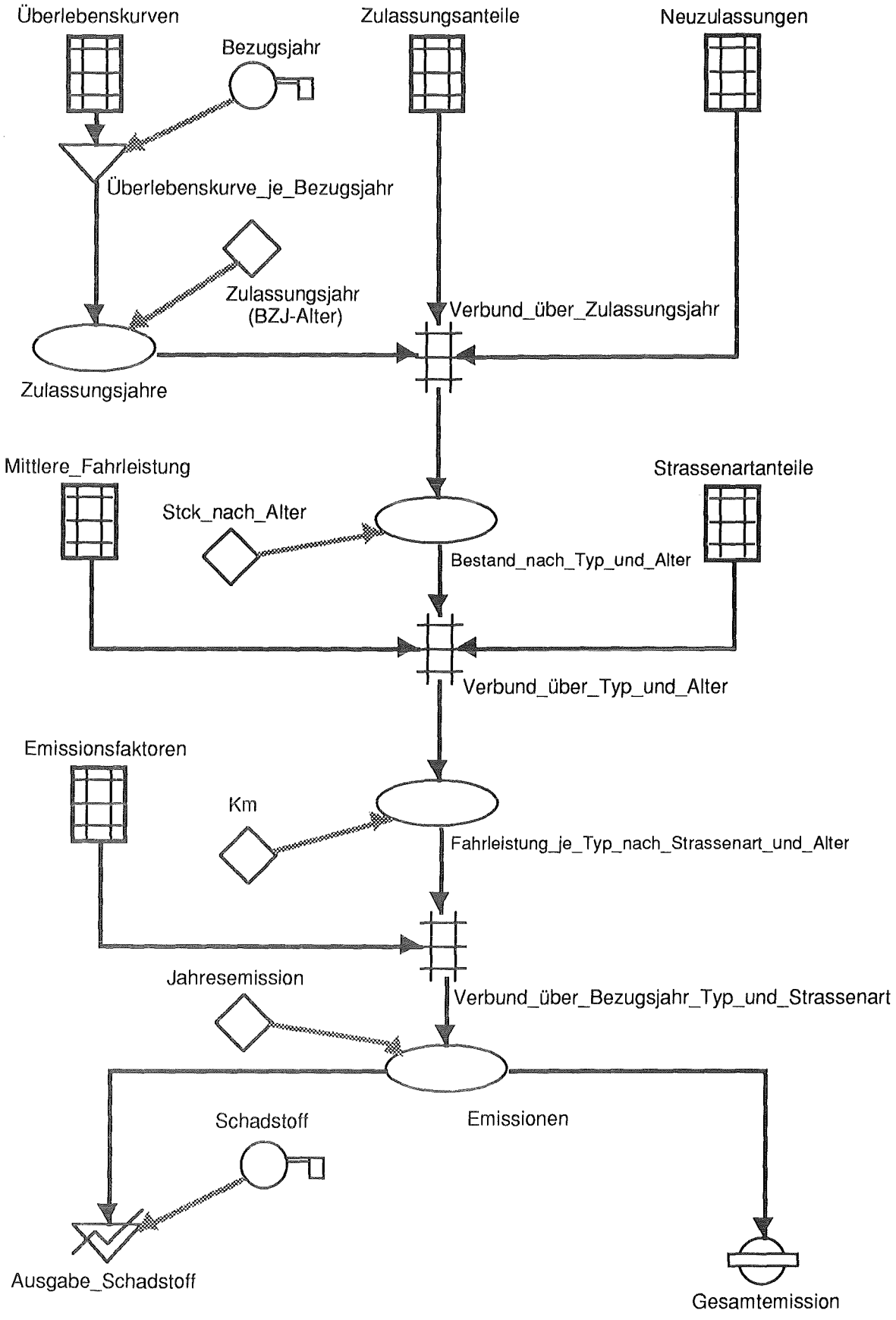
Grützner, R., Häuslein, A., Page, B. (1993)
Softwarewerkzeuge für die Umweltmodellierung und -simulation
In: Umweltinformatik, Informatikmethoden für Umweltschutz und Umweltforschung
Oldenbourg Verlag, München

Jaeschke, A., Kämpke, T., Page, B., Radermacher, F.J. (1993)
Informatik für den Umweltschutz
Springer Verlag, Berlin Heidelberg

Niederle, W., (1991)
"Rapid Prototyping" von Verkehrsemissionsmodellen
In: Hälker, M., Jaeschke, A.(1991) Informatik für den Umweltschutz.
Springer Verlag, Berlin Heidelberg

Zehnder, C.A. Informationssysteme und Datenbanken (1987)
B.G. Teubner Verlag, Stuttgart

Anhang: Modelldiagramm



Teilnehmerliste

AK5 : Wekzeuge für Simulation und Modellbildung in Umweltanwendungen

Witzenhausen, 28.10 - 29.10.93

Dr. Joachim Benz Universität Gesamthochschule Kassel
Fachbereich 11
Nordbahnhofstr. 1a
37213 Witzenhausen
email: benz@wiz.uni-kassel.de

Christian Bohley Universitätsrechenzentrum Halle
Weinbergerweg 17
06120 Halle/Saale

Evgeni Dimitrov Universität Rostock
FB Informatik
AG Modellierung/Simulation
Albert-Einstein-Str. 1
18501 Rostock

Henning Freese Universität Hamburg
Fachbereich Informatik
Vogt-Koelln-Str. 30
22527 Hamburg

Norbert Grebe Universität Passau
Lehrstuhl für Operations Research und
Systemtheorie
Innstr. 33
94032 Passau
email: grebe@moni.fmi.uni-passau.de

Prof. Dr. Rolf Grützner Universität Rostock
FB Informatik
AG Modellierung/Simulation
Albert-Einstein-Str. 1
18501 Rostock

Klaus-F. Kaal TIMOlogic GmbH
Fuchsweg 186
78351 Ludwigshafen/Bodensee

Dr. Hubert Keller Kernforschungszentrum Karlsruhe GmbH
Institut für Angewandte Informatik (IAI)
Postfach 3640
76021 Karlsruhe
email: hubert.keller@idt.kfk.d400.de

Markus Kuna Universität Duisburg
FB Wirtschaftsinformatik
Kastanienstr.7
47269 Duisburg

Prof. Dr. Bernd Page Universität Hamburg
Fachbereich Informatik
Vogt-Koelln-Str. 30
22527 Hamburg

email: page@rz.informatik.uni-hamburg.d400.de

Dr. Wolfgang Paul Forschungsanstalt für Landwirtschaft (FAL)
Bundesallee 50
38116 Braunschweig

Dr. Michael Ruge Siemens AG
Technikfolgenforschung
ZFE BT SE 2
81730 München
email: ruge@sioux.zfe.siemens.de

Christof Scholz Universität Gesamthochschule Kassel
Fachbereich 11
Nordbahnhofstr. 1a
37213 Witzenhausen
email: scholz@wiz.uni-kassel.de

Jürgen Seidel Universität Hamburg
Fachbereich Informatik
Vogt-Koelln-Str. 30
22527 Hamburg

Kirsten Sonnenberger BEC-Vertriebsgesellschaft mbH
TZW
Richard Wagner Str. 31
18119 Warnemünde

Uwe Stickel TIMOlogic GmbH
Fuchsweg 186
78351 Ludwigshafen/Bodensee

Dr. Ing. Klaus Weigelt BEC-Vertriebsgesellschaft mbH
TZW
Richard Wagner Str. 31
18119 Warnemünde