DEVS Standardization Group

Seminar „Das virtuelle Labor"
Wintersemester 2007/2008
Silvester Klement

Gliederung:

Abstract
1. Motivation
2. Ziel von DEVS
3. Aufbau von DEVS
   - 3.1 Grundprinzip
   - 3.2 Aufbau der Grundmodelle
   - 3.3 Gekoppelte Modelle
4. Anwendungsbeispiele
   - 4.1 Simulation von Flächenbränden
   - 4.2 Modellierung und Simulation des zellulären Stoffwechsels von Mitochondrien
5. Was soll standardisiert werden?
6. Zusammenfassung
7. Quellen

Abstract

Bisherige Simulationsverfahren verwendeten direkt auf dieses Verfahren zugeschnittene Modelle und Simulatoren, was den Nachteil hatte, dass diese schlecht integrier- und wiederverwendbar waren. DEVS (Discrete Event System Specification) hat eine andere Herangehensweise, indem es die Modelle speziell darauf auslegt, wiederverwendbar zu sein und die Simulatoren integrierbar und dadurch verteilt auf mehreren Rechnern ausführbar macht. Die DEVS Standardization Group hat es sich nun zum Ziel gesetzt, diese Simulations- und Modellierungsmethode zu standardisieren, um u.a. die Interoperabilität mit anderen Standards wie z.B. XML oder CORBA, sowie von DEVS- mit Nicht-DEVS-Modellen zu erreichen. Diese Arbeit legt dar, was die Motivation für die Entwicklung von DEVS war, was das Ziel dieser Entwicklung war, wie der Aufbau von DEVS aussieht und es werden Anwendungsbeispiele genannt.

1. Motivation

Die bisherige Herangehensweise an Simulationsexperimente war die, dass ein abzubildendes System analysiert und dann für dieses spezielle System eine Abbildung in Form eines Modells geschaffen wurde. Der einzige Fortschritt, der hierbei gemacht wurde, war der, den die Technik in Bezug auf die Leistung der Rechner, Netzwerk- oder Grafikkarten etc. vollzog. Es wurden lediglich Lösungen für spezielle Problemstellungen gefunden, was dazu führte, dass die Modelle, Simulatoren und Experimente alle sehr spezifisch, sowie schlecht integrier- und wiederverwendbar waren. Hier setzt nun DEVS an, ein Simulations- und Modellierungsverfahren, das mit einem Hauptaugenmerk auf Integrierbarkeit, Wiederverwendbarkeit und Interoperabilität konzipiert wurde. Doch obwohl diesem Verfahren ein mathematischer Formalismus zugrundeliegt, was es prinzipiell unabhängig von verschiedenen Programmiersprachen und Hardwareplattformen macht, gibt es verschiedene Entwicklungsgruppen weltweit [Link1], was die Implementation kompliziert werden lässt und das Wiederverwenden und Austauschen von DEVS Modellen erschwert. Das Ziel der DEVS Standardization Group ist

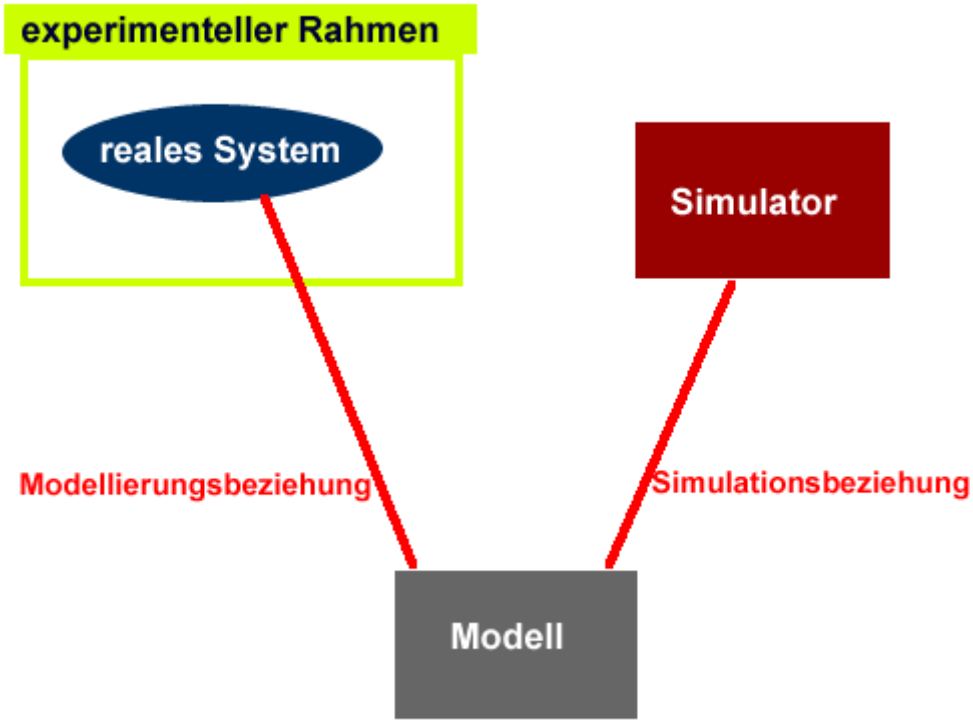
es nun, Standards für eine durch Computer ausführbare Repräsentation des DEVS Formalismusses zu schaffen.

2. Ziel von DEVS

DEVS steht für Discrete EVent System Specification und wurde 1976 von Bernard P. Zeigler konzipiert [BeZe76]. Hierbei werden die Modelle und Simulationen als eigene Entitäten oder Dinge (englisch: „entities") gehandhabt. Die Modelle sind systematisch dafür ausgelegt, wiederverwendbar zu sein, wie exemplarisch unter [Link2] zu sehen ist. Hier werden Repositorien, d.h. ein System in dem digitale Objekte gespeichert werden und verfügbar sind [Link3], erstellt und gewartet, in denen die Modelle mit ihren dazugehörigen Informationen abgelegt werden. Zudem ist es möglich, die Modelle hierarchisch und modular zu entwickeln. Die Simulatoren sind darauf ausgelegt, entweder, wie bisher, an einem einzigen Rechner ausgeführt zu werden, oder verteilt auf mehreren, was einen Performancegewinn, gerade bei komplexeren Simulationensexperimenten, möglich macht.

3. Aufbau von DEVS

3.1 Grundprinzip



[Abbildung 1]

Die Herangehensweise von DEVS an Modellierung und Simulation ist wie in Abbildung 1 zu sehen, dass zuerst einmal aufgeteilt wird in drei verschiedene Bereiche: das reale System, das Modell und schließlich der Simulator.
Das reale System, existent oder nur angenommen, ist das Objekt, das durch die Simulation abgebildet werden soll. Es wird als Datenquelle behandelt und befindet sich in einem experimentellen Rahmen, der festhält, wie die Ziele des Modellierers die Konstruktion, Experimentierung mit und Überprüfung des Modells beeinflussen.
Das Modell ist eine Liste aus Instruktionen, die dazu dienen, Daten zu kreieren, die denen aus dem realen System entsprechen. Es gilt als ein zulässiges Modell, solang es

Daten produziert, die den Daten, die das reale System produziert, ähnelt. Das Verhalten des Modells ist eine Liste aller Daten, die durch seine Ausführung entstehen können.
Der Simulator führt die Instruktionen des Modells aus, um dessen Verhalten zu erzeugen. Diese Objekte sind durch zwei Beziehungen verknüpft. Zum einen der Modellierungsbeziehung zwischen realem System und Modell, die angibt, wie gut das Modell das System abbildet. Zum Anderen der Simulationsbeziehung zwischen Modell und Simulator, die angibt, wie genau der Simulator in der Lage ist, die Instruktionen des Modells auszuführen.

3.2 Aufbau der Grundmodelle

Die Struktur des Modells wird nun in einem mathematischen Formalismus ausgedrückt, der definiert, wie neue neue Werte für Variablen erzeugt werden und wann diese inkraft treten. Ein wichtiger Aspekt dieses Formalismusses ist, dass die Zeitintervalle zwischen diesen Variablenänderungen, sogenannten Events, variabel sind.
Innerhalb des DEVS Formalismus werden zum Einen die Grundmodelle, aus denen größere Modelle gebaut werden können und zum Anderen wie diese miteinander hierarchisch verbunden sind spezifiziert.
Ein Modell selbst besitzt Eingangs- und Ausgangsports, durch die jegliche Interaktion mit der Umgebung stattfindet. Events bestimmen die Werte die an diesen Ports auftreten. Wenn externe Events außerhalb des Modells zum Eingangsport gelangen, muss die Modellbeschreibung bestimmen, wie das Modell darauf reagiert. Durch eine Variable festgelegte Intervall-Events innerhalb des Modells verändern dessen Zustand und werden als Events an den Ausgangsports an die anderen Modellkomponenten weitergegeben.
Im Detail enthält ein Grundmodell die folgenden Informationen:

- ein Satz Eingangsports, durch die externe Events ankommen
- ein Satz Ausgangsports, durch die externe Events ausgesendet werden
- ein Satz Zustandsvariablen und –parametern, in der Regel gibt es zwei Zustandsvariablen, „phase" und „sigma" (solange kein externes Event stattfindet, bleibt das System in der aktuellen „phase" für die Zeitdauer „sigma")
- die Zeitfortschrittsfunktion, die das Timing von internen Zustandsübergängen kontrolliert, wenn die „sigma"-Zustandsvariable vorhanden ist, wird diese zurückgegeben
- die interne Zustandsübergangsfunktion, die angibt, in welchen Zustand das System wechselt, wenn die von der Zeitfortschrittsfunktion zurückgegebene Zeit erreicht wird
- die externe Zustandsübergangsfunktion, die angibt, wie das System seinen Zustand ändert, wenn ein Event am Eingangsport vorliegt, mit dem Effekt, dass das System in eine neue „phase" und ein neues „sigma" gelangt und daher für einen internen Zustandsübergang vorgesehen ist. Der nächste Zustand wird anhand des aktuellen Zustandes des Eingangsports, dem Wert des externen Events und der Zeit die bisher im aktuellen Zustand verstrichen ist, berechnet.
- die zusammenfließende Zustandsübergangsfunktion, die angewandt wird, wenn ein Event am Eingangsport zur selben Zeit auftritt wie ein geplanter interner Zustandsübergang, standardmäßig wird dann zuerst die interne Zustandsübergangsfunktion ausgeführt und anschließend die externe auf den aus ersterem resultierenden Zustand.
- die Ausgabefunktion, die eine externe Ausgabe erzeugt, bevor ein interner Zustandsübergang stattfindet.

3.3 Gekoppelte Modelle

Grundmodelle können im DEVS Formalismus gekoppelt werden, um ein gekoppeltes Modell zu erzeugen. Ein gekoppeltes Modell sagt aus, wie mehrere Komponentenmodelle verbunden werden müssen um ein neues Modell zu erzeugen. Dieses Modell selbst kann innerhalb des DEVS Formalismusses als äquivalent zu einem Grundmodell ausgedrückt

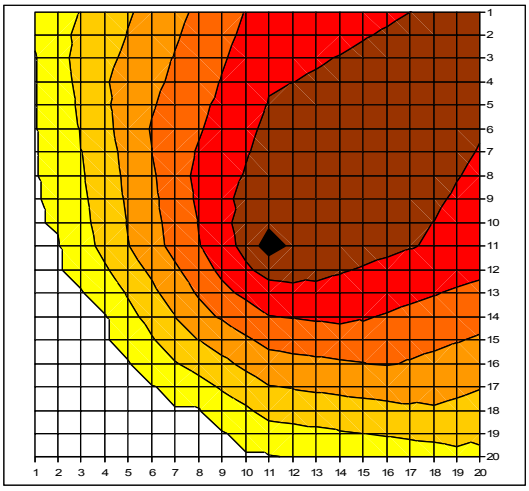
werden und kann dadurch eine Komponente in einem größeren gekoppelten Modell sein, was eine hierarchische Konstruktion zur Folge hat.
Ein gekoppeltes Modell enthält folgende Informationen:

- einen Satz Komponenten
- einen Satz Eingangsports, durch den externe Events empfangen werden
- einen Satz Ausgangsports, durch den externe Events ausgesandt werden
- eine externe Eingangskopplung, die die Eingangsports des gekoppelten Modelles mit den Eingangsports der Komponenten verbindet, sodass Eingaben in das gekoppelte Modell direkt an die gewünschten Komponenten weitergeleitet werden
- eine externe Ausgangskopplung, die die Ausgangsports der Komponenten mit dem Ausgangsport des gekoppelten Modells verbindet, sodass wenn eine Ausgabe von einer Komponente erzeugt wird, diese von einem gewünschten Ausgangsport extern versandt werden kann
- einer internen Kopplung, die Ausgangsports von Komponenten mit Eingangsports von anderen Komponenten verbindet, wodurch eine Ausgabe, zusätzlich zum Ausgangsport des gekoppelten Modells, an die Eingangsports der anderen Komponenten gesendet werden kann

4. Anwendungsbeispiele

4.1 Simulation von Flächenbränden [LeNtBiKh2006]

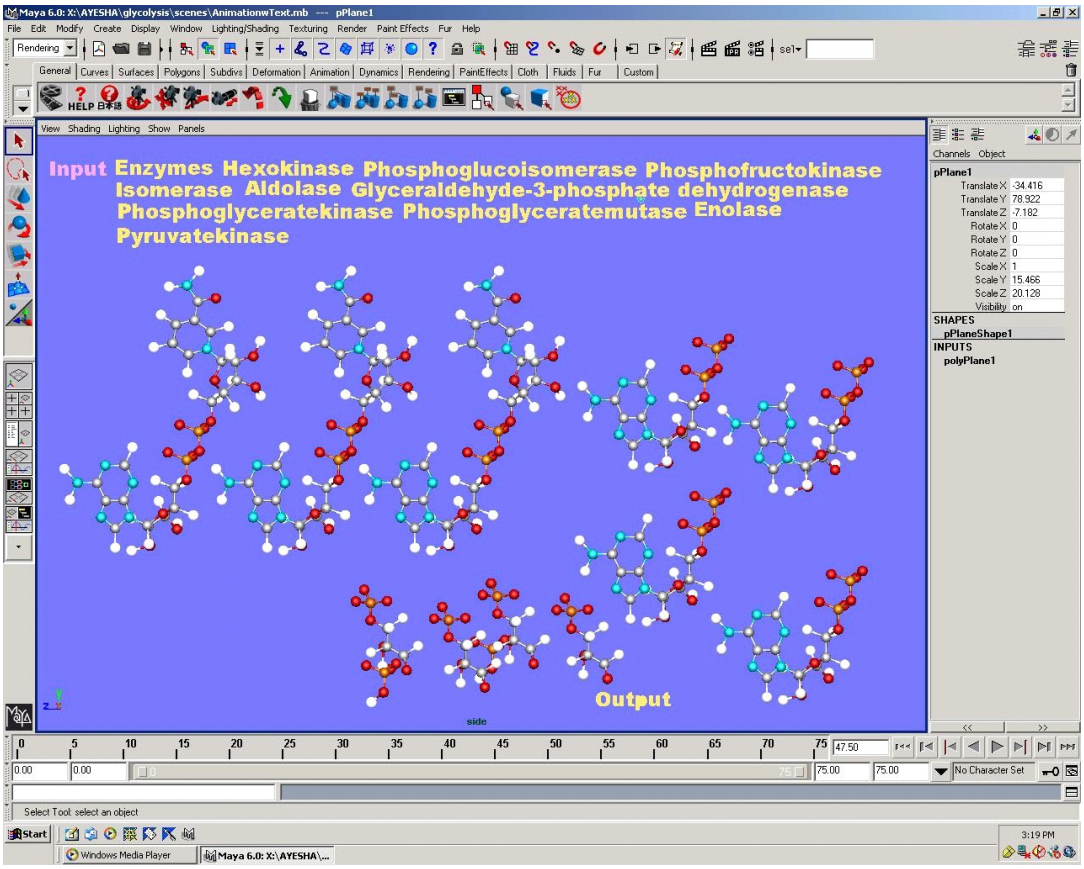
Flächenbrände und die mit ihnen einhergehende Zerstörung zu simulieren soll der Feuerwehr helfen, die Ausbreitung der Feuermassen einzuschätzen und damit deren Eindämmung zu vereinfachen. Da diese Ausbreitung ein komplexer Prozess ist, werden Simulationsmodelle benötigt, die sowohl Zeit als auch Raum beinhalten, was eine große Menge Daten erfordert. Hierfür wird eine spezielle Variante von DEVS namens Cell-DEVS verwandt, wobei der Wald in kleine, Zellen (“Cells”) genannte, Bereiche unterteilt wird. Diese Bereiche werden im Computer als Zellbereiche (“cell-space”) dargestellt, was deren geografische Beziehungen aufrecht erhält. Die Ausbreitung des Feuers wird nun zuerst basierend auf einem mathematischen Feuerausbreitungsmodell innerhalb einer Zelle berechnet, und schließlich als Ansteckungsprozess mehrere Zellen übergreifend. Das mathematische Modell von Rothermel [Rothermel72] erlaubt es nun, eine eindimensionale, maximale Vorwärtsausbreitungsrichtung und –geschwindigkeit zu berechnen. Die Ausbreitung in alle anderen Richtungen lässt sich daraus ableiten. Abbildung 2 zeigt das Ergebnis einer solchen Simulation, wobei das schwarze Viereck in der Mitte der Abbildung der Ausgangspunkt des Feuers ist, das sich durch Wind aus der südwestlichen Richtung in die nordöstliche Richtung ausbreitet. Der Farbübergang vom bräunlichen ins gelbe symbolisiert den Verlauf der Ausbreitung des Feuers.



[Abbildung 2]

4.2 Modellierung und Simulation des zellulären Stoffwechsels von Mitochondrien [RoDjGaWaToMu2005]

Aufgrund seiner hierarchischen Struktur ist DEVS gut geeignet, um biologische hierarchische Systeme wie z.B. Zellen zu beschreiben. Die Eventbasierte Natur von DEVS macht es nun möglich, die in einer Zelle auftretenden chemischen Prozesse adäquat abzubilden. In diesem speziellen Fall handelt es sich um den zellulären Stoffwechsel von Mitochondrien, die im Stoffwechsel des Körpers mehrere wichtige Rollen spielen. So werden mehrere genetische Erkrankungen, z.B. Diabetes, Taubheit, Alzheimer oder Parkinson, von einer Mutation der Mitochondrien DNS verursacht. Mitochondrien sind kleine Organelle, d.h. Bereiche einer Zelle mit einer besonderen Funktion, die dafür verantwortlich sind, Nährstoffe in das Molekül Adenosintriphosphat umzuwandeln, das die Zellen mit Energie versorgt. Wenn die Zellen Energie verbrauchen, verwandelt sich Adenosintriphosphat zurück in Adenosindiphosphat, was in den Mitochondrien dann erneut verwandt wird, um Adenosintriphosphat herzustellen. In den Mitochondrien findet also ein ständiger Stoffwechsel statt, der, chemisch betrachtet, in mehreren Schritten vollzogen wird. Es ist nun möglich, jeden dieser Schritte nachzuvollziehen, in [Abbildung 3] ist z.B. Schritt Nummer 6 von 10 dieses Prozesses zu sehen. Im oberen Bereich der Grafik ist der aktuelle Input zu sehen, was in Punkt 3.2 dieser Arbeit als Event am Eingangsport bezeichnet wurde. Der Vorteil der Simulation dieses Stoffwechsels durch DEVS besteht darin, dass die einmal erstellten Modelle wiederverwendet oder in komplexere Simulationen eingebaut werden können.



[Abbildung 3]

5. Was soll standardisiert werden?

Daher soll ein Standard für DEVS-basierte Tools und Erweiterungen entwickelt werden, der der SISO (Simulation Interoperability Standards Organization) vorgelegt und zum IEEE Standard werden soll.
Hauptaufgaben der Standardisierung sind hierbei die Beziehung zu anderen Standards wie z.B. HLA (hla.dmso.il), CORBA (www.omg.org/corba), XML (www.w3.org) oder Modelica (www.modelica.org), die Interoperabilität zwischen DEVS und Nicht-DEVS Modellen sowie die Standardisierung grundsätzlicher DEVS Modellierungskonstrukte.

6. Zusammenfassung

In dieser Arbeit wurde dargelegt, was die Motivation für die Entwicklung von DEVS als Simulations- bzw. Modellierungsverfahren war, welches Ziel damit verfolgt wird, wie dessen Aufbau aussieht und exemplarisch gezeigt, wie es eingesetzt werden kann. Außerdem wurden die Ziele und Gründe der Standardisierung von DEVS aufgezeigt.
Offen bleibt, wann und wie die Standardisierung abgeschlossen wird, sowie ob und in welcher Form Änderungen am DEVS Formalismus im Zuge dieses Prozesses vorgenommen werden. Da zu diesem Thema bisher nur ein bei der SISO eingereichter Antrag vorliegt [Link4], bleibt hier nur die Spekulation. Wünschenswert wäre eine Standardisierung auf jeden Fall, da die Interoperabilität und Wiederverwendbarkeit von (DEVS-)Simulationsmodellen vor dem Hintergrund globaler Forschungsvernetzung, sowie Ressourcen- und Zeiteinsparung ein wichtiger Schritt wäre.

7. Quellen/Abbildungen/Literaturangaben:

Quellen:
http://www.cs.gsu.edu/DEVS/ 6.11.2007 21:30Uhr
http://www.sce.carleton.ca/faculty/wainer/standard/ 6.11.2007 21:30Uhr
http://www.acims.arizona.edu/SOFTWARE/devsjava_licensed/CBMSManuscript.zip 6.11.2007 21:30Uhr

Links:
[Link1] http://www.sce.carleton.ca/faculty/wainer/standard/tools.htm 6.11.2007 21:30Uhr
[Link2] http://www.sce.carleton.ca/faculty/wainer/wbgraf/samplesmain_1.htm 7.11.2007 23:18Uhr
[Link3] http://www.uibk.ac.at/ub/dea/docs/muehlberger_wuv.pdf 7.11.2007 23:18Uhr
[Link4] http://www.sce.carleton.ca/faculty/wainer/standard/finalReport05.pdf 6.11.2007 21:30Uhr

Abbildungen:
[Abbildung 1] selbst angefertigt
[Abbildung 2] aus
http://www.canarie.ca/conferences/platforms/presentations/wainer.ppt S.17

Literaturangaben:
[BeZe76] Bernard P. Zeigler: Theory of Modeling and Simulation John Wiley 1976

[RoDjGaWaToMu2005] Roxana Djafarzadeh, Gabriel Wainer, Tofy Mussivand: DEVS Modeling and Simulation of the Cellular Metabolism by Mitochondria http://www.sce.carleton.ca/faculty/wainer/papers/DEVS-Mito.pdf 7.11.2007 23:19Uhr

[LeNtBiKh2006] Lewis Ntaimo, Bithika Khargharia: Two-Dimensional Fire Spread Decomposition in Cellular DEVS Models 2006 http://ie.tamu.edu/people/faculty/Ntaimo/personal_web/Papers/NtaimoBithikaRevised020206.pdf 6.11.2007 21:30Uhr

[Rothermel72] Rothermel, R. 1972. \A Mathematical Model for Predicting Fire Spread in Wildland Fuels," Research Paper INT-115. Ogden, UT: U.S. Dept. of Agriculture, Forest Service, Intermountain Forest and Range Experiment Station.