

Lineare Prozessmodelle – Vom Ideal zur praxistauglichen Methode

Guido Schimm

OFFIS
Escherweg 2
D-26121 Oldenburg
Schimm@offis.de

Abstract: Lineare Prozessmodelle werden heute in zahlreichen Anwendungsbereichen eingesetzt, da sie sich durch eine Reihe von vorteilhaften Eigenschaften auszeichnen. Hierzu gehören vor allem Deadlock-Freiheit, Adaptierbarkeit und Struktur der Modelle. Lineare Prozessmodelle werden aber auch mit einigen Defiziten verbunden und daher eher als Ideal denn als praxistaugliche Methode angesehen. Bei diesen Defiziten handelt es sich primär um die Modellierung nicht-linearer Steuerflüsse sowie das ausschließliche Top-down-Vorgehen bei der Modellierung. Im vorliegenden Beitrag wird zunächst ein Meta-Prozessmodell für lineare Prozessmodelle vorgestellt. Hierbei wird auf die Bausteine, eine Algebra und Normalformen der Modelle eingegangen. Für die aufgezeigten Defizite linearer Prozessmodelle werden anschließend Lösungsansätze aufgezeigt, durch welche ein Beitrag zur Erhöhung der Praxistauglichkeit linearer Prozessmodelle geleistet werden soll.

1 Einleitung

In vielen Anwendungs- und Forschungsbereichen werden heute Prozessmodelle eingesetzt. In Abhängigkeit vom jeweils verwendeten Meta-Prozessmodell besitzen Prozessmodelle nicht nur ganz unterschiedliche Formen und Eigenschaften, sondern sie können sich auch durch eine unterschiedliche oder nicht hinreichend definierte Semantik der in den einzelnen Meta-Prozessmodellen verwendeten Modellelemente unterscheiden. Bei vielen Metamodellen der Prozessmodellierung, wie bspw. den verbreiteten Diagrammsprachen, ist jedoch eine Konvergenz zur Verwendung von Bausteinen mit definierter Semantik und linearen Kontrollstrukturen zu beobachten [z.B. Rum99]. Häufig handelt es sich um Modellierungskonventionen oder Richtlinien, die Empfehlungscharakter besitzen und nicht Bestandteil des Metamodells sind. Aber auch präzise Forderungen von Eigenschaften der Prozessmodelle wie z.B. Soundness für Workflow-Netze [Aal98, Aal00] sind zu finden. Lineare Prozessmodelle sind mit diesen Modellen vergleichbar. Sie werden aus Bausteinen zusammengesetzt, die den in Diagrammsprachen verwendeten Netzfragmenten entsprechen, und besitzen einen

linearen Steuerfluss. Vergleichbare Prozessmodelle sind auch unter der Bezeichnung hierarchische oder strukturierte Prozessmodelle zu finden [vgl. Rit00, KHB00].

Lineare Prozessmodelle besitzen die Soundness-Eigenschaft, d.h. sie sind frei von Deadlocks [Aal00]. Die sonst notwendigen Zyklen der Verifikation bzw. Simulation der Modelle vor ihrem Einsatz können daher entfallen. Weitere vorteilhafte Eigenschaften sind die leichte Adaptierbarkeit der Prozessmodelle, die Möglichkeit, die Ausprägung einzelner Prozessbausteine erst zur Laufzeit bestimmen zu müssen, sowie die Verwendung von Prozessschablonen [Sch00].

Als Defizit linearer Prozessmodelle wird in der Literatur die Einschränkung von Ausdrucksfreiheit und Ausdrucksmächtigkeit beschrieben [KHB00, Rit00]. Neben einer rein dekompositionellen Modellierung, die eine hohe Abstraktionsleistung vom Modellierer fordert, erscheint die Modellierung nicht-linearer paralleler Steuerflüsse nicht möglich. Aktuelle Forschungsarbeiten haben Lösungsansätze für diese Probleme hervorgebracht und umgesetzt. Dieser Beitrag zeigt diese Lösungsansätze auf und soll damit zur Erhöhung der Praxistauglichkeit linearer Prozessmodelle beitragen.

Im zweiten Abschnitt werden zunächst die Bausteine linearer Prozessmodelle vorgestellt. Algebren und Normalformen für lineare Prozessmodelle werden im dritten Abschnitt angerissen. Der vierte Abschnitt beschreibt einen Ansatz für die Modellierung nicht-linearer Steuerflüsse in linearen Prozessmodellen. Den Lösungsansatz zur Kombination von Top-down- und Bottom-up-Modellierung adressiert ein Werkzeug zur Modellierung, Analyse und Simulation linearer Prozessmodelle, das im fünften Abschnitt vorgestellt wird. Der Beitrag schließt mit einem Ausblick.

2 Bausteine linearer Prozessmodelle

Die Bausteine linearer Prozessmodelle sind Blöcke. Diese Blöcke können als Netzfragmente aufgefasst werden, die eine bestimmte Semantik besitzen. Sie werden in der Modellierungspraxis häufig als Entwurfskonstrukte verwendet [Bas98].

Alle Blöcke sind in der Außensicht identisch: Jeder Block besitzt für den Steuerfluss genau einen Eintritts- und einen Austrittspunkt. Blöcke und die aus ihnen erstellten Prozessmodelle werden daher in Anlehnung an die lineare/strukturierte Programmierung als linear bezeichnet.

Blöcke können zunächst in Operanden und Operatoren differenziert werden. Operanden sind Aktivitäten, Subprozesse und generische Blöcke. Basisoperatoren sind die sequenzielle, alternative und parallele Komposition sowie die Iteration. Der syntaktische Unterschied zwischen Operanden und Operatoren besteht zunächst darin, dass Operatoren Blöcke sind, die weitere Blöcke enthalten.

Die Modellierung von Prozessen erfolgt durch die beliebig tiefe Verschachtelung von Operatoren, die durch das Einsetzen von Operanden, die selbst keine Blöcke enthalten, terminiert wird. Es resultieren Prozessmodelle, die in einen linearen Steuerfluss die dem

Prozess zugehörigen Aktivitäten einbetten. Jedem Prozessmodell liegt aufgrund dieser Verschachtelung seiner Blöcke eine Baumstruktur zugrunde, dessen Wurzel ein beliebiger Block ist und dessen Blätter die Aktivitäten des Prozesses sind. Abbildung 1 gibt einen Überblick über die einzelnen Blocktypen und die zwischen ihnen bestehenden Beziehungen in UML-Notation.

Der wichtigste Operand ist die Aktivität (Activity). Er wird als Ausführungs- und Kontrollanweisung für eine bestimmte Instanz einer Funktion des betrachteten Systems interpretiert. Aktivitäten können hinsichtlich der betreffenden Funktion weiter differenziert werden. Üblich ist bspw. eine Unterscheidung manueller und automatisierter Aktivitäten. Die Menge der Aktivitäten eines Prozessmodells wird als diskretes Prozessmodell bezeichnet. Mit Hilfe von Subprozess-Operanden (Process) werden Prozesse im Rahmen der vertikalen Hierarchisierung als Operanden verwendet. Diese Operanden können durch das Einsetzen des referenzierten Prozesses substituiert werden. Generische Blöcke (Generic) sind Operanden, die während der Erstellung von Prozessmodellen Verwendung finden. Sie dienen als Platzhalter für beliebige Blöcke und erlauben es, Prozessfragmente komplexer Steuerflüsse als Schablonen einzusetzen.

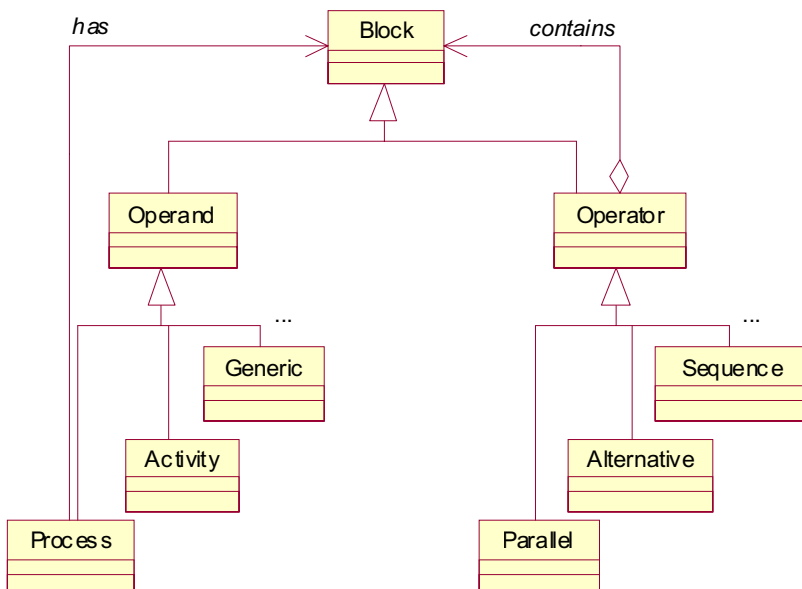


Abbildung 1: Bausteine linearer Prozessmodelle

Die sequenzielle Komposition (Sequence) steht für die Ausführung einer Menge von Blöcken in einer definierten Reihenfolge. In vielen Diagrammsprachen wird die sequenzielle Komposition durch das Verbinden der Knoten durch gerichtete Kanten dargestellt. In linearen Prozessmodellen ist die sequenzielle Komposition ein Block, der die exakte Reihenfolge der in ihm enthaltenen Blöcke festlegt.

Die parallele Komposition (Parallel) bezeichnet wie die sequenzielle Komposition ein Prozesskonstrukt, das alle in ihm enthaltenen Blöcke ausführt. Im Gegensatz zur Sequenz werden die in den Steuerfluss eingebetteten Blöcke hier jedoch nebenläufig ausgeführt. In Diagrammsprachen wird diese Konstrukt durch eine Aufspaltung des Steuerflusses und der korrespondierenden Verschmelzung modelliert. Die Verschmelzung ist wichtig, da durch sie eine Synchronisation der mehrfachen Steuerflüsse erreicht und die Forderung nach exakt einem Austrittspunkt gewährleistet wird.

Die alternative Komposition (Alternative) bettet wie die parallele Komposition jeden enthaltenen Block in einen separaten Steuerfluss ein. Der Unterschied zu den beiden vorangegangenen Basisoperatoren besteht jedoch darin, dass nur genau einer der enthaltenen Blöcke ausgeführt wird. Die parallele Komposition ist also eine Selektion aus mehreren Alternativen.

Die Iteration ist ein einstelliger Operator. An ihrer Stelle wird in Prozessalgebren die Rekursion verwendet. Die Iteration beinhaltet wie die alternative Komposition eine Entscheidung. Im Gegensatz zur alternativen Komposition geht es bei der Iteration jedoch nicht darum, einen Block aus einer Menge von Blöcken auszuwählen, sondern die Auswahl bzw. wiederholte Auswahl genau eines Blocks zu entscheiden. Die Iteration ist ein Konstrukt der Modellierungspraxis, das Sequenzen gleicher Blöcke zusammenfasst und die vorstehenden Operatoren ergänzt.

Weitere Operanden und Operatoren können dem Metamodell hinzugefügt werden. So kann z.B. ein Operatoren für ein Interleaving von Blöcken verwendet werden. Wichtig hierbei ist die Festlegung der Semantik dieser Operatoren. Im Falle des Interleaving steht der neue Operator für ein komplexes, aus den Basisoperatoren zusammengesetztes Prozessfragment. Durch die Verwendung abgeleiteter Operatoren kann eine Vereinfachung der Modellierung erreicht werden, da sie komplexe Steuerflüsse aggregieren.

Lineare Prozessmodelle können als Graphen, Terme oder Bäume dargestellt werden. Die Darstellung als Term ist sehr kompakt und eignet sich daher, um bspw. Umformungen der Modelle zu notieren. Als interne Repräsentationsform für Implementierungen haben sich Bäume etabliert. Graphen eignen sich hingegen besonders für eine anschauliche Visualisierung der Prozessmodelle.

3 Algebren und Normalformen

Für das Metamodell linearer Prozessmodelle können Algebren angegeben werden. Die Axiome dieser Algebren dienen der Umformung von Prozessmodellen und der Ableitung von Termersetzungssystemen. Außerdem können sie zum Beweis der Äquivalenz von Prozessen herangezogen werden. Eine Algebra für lineare Prozessmodelle, welche die vorgestellten Basisoperatoren verwendet, wird in der Tabelle 1 vorgestellt. Der Operator der sequentiellen Komposition wird mit S bezeichnet, der für die parallele Komposition mit P und der für die alternative Komposition mit A. Die Operatoren werden als n-stellige

Präfix-Operatoren eingesetzt. Die Iteration wird mit L bezeichnet und ist ein unärer Operator. Die Variablen sind allquantifiziert. Die Algebra enthält kein Axiom für die Linksdistributivität der alternativen über die sequentielle Komposition. Dieses Axiom ist nur bei einer retrospektiven oder antizipativen Verwendung eines Prozessmodells zulässig, da es zur Verschiebung von Entscheidungszeitpunkten führt [BaWe90; Ritt98].

sorts:	Activities, Processes $\text{Activities} \subset \text{Processes}$
consts:	Activities, Sub Processes, Generics
ops:	$A, S, P: \text{Proc.} \times \dots \times \text{Proc.} \rightarrow \text{Proc.}$
eqns:	$\begin{aligned} \text{A1:} \quad & A_1(x_1, \dots, x_n) = \dots = A_n(x_n, \dots, x_1) \\ \text{A2:} \quad & A(\dots, x_1, \dots, x_n, \dots) = A(\dots, A(x_1, \dots, x_n), \dots) \\ \text{A3:} \quad & A(x, \dots, x) = x \\ \text{A4:} \quad & S(A(x_1, \dots, x_n), y_1, \dots, y_m) = \\ & A(S_1(x_1, y_1, \dots, y_m), \dots, S_n(x_n, y_1, \dots, y_m)) \\ \text{A5:} \quad & S(\dots, x_1, \dots, x_n, \dots) = S(\dots, S(x_1, \dots, x_n), \dots) \\ \text{A6:} \quad & P_1(x_1, \dots, x_n) = \dots = P_n(x_n, \dots, x_1) \\ \text{A7:} \quad & P(A(x_1, \dots, x_n), y_1, \dots, y_m) = \\ & A(P_1(x_1, y_1, \dots, y_m), \dots, P_n(x_n, y_1, \dots, y_m)) \\ \text{A8:} \quad & P(\dots, x_1, \dots, x_n, \dots) = P(\dots, P(x_1, \dots, x_n), \dots) \\ \text{A9:} \quad & L(x) = S(x, \dots, x) \end{aligned}$

Tabelle 1 : Algebra linearer Prozessmodelle

Neben Algebren werden auch Normalformen für lineare Prozessmodelle verwendet. Zwei dieser Normalformen sind die Kopf-Normalform (KNF) und die disjunktive Normalform (DNF). Die KNF eines linearen Prozessmodells besteht aus einer alternativen Komposition, die Sequenzen einbettet, deren Operanden stets vor deren Operatoren liegen. Ein Prozessmodell wird in seine KNF durch Anwendung eines Termersetzungssystems überführt, das aus der in Tabelle 1 vorgestellten Algebra abgeleitet werden kann. Die Bezeichnung der DNF wurde von [Ritt98] eingeführt und erfolgte in Anlehnung an die disjunktive Normalform der Aussagenlogik. Ein Prozessmodell in DNF besteht aus einer alternativen Komposition, die alle möglichen sequenziell-parallelen Kompositionen der Operanden des Prozesses umfasst. Ein Prozessmodell wird in seine DNF durch Anwendung von Termersetzungssystemen überführt, welche die volle Distributivität der sequentiellen Komposition über die alternative Komposition unterstützen. Zur Bildung eines solchen Termersetzungssystems wird die oben vorgestellte Algebra um die fehlende Linksdistributivität erweitert.

4 Modellierung nicht-linearer Steuerflüsse

Lineare Prozessmodelle scheinen bestimmte Prozessstrukturen nicht abbilden zu können. Es handelt sich hierbei um Strukturen nicht-linearer Steuerflüsse, die den Ausbruch aus einem linear ineinander geschachtelten Steuerfluss erfordern. Das lineare Modell verlangt, dass eine Verzweigung, die innerhalb einer anderen Verzweigung geöffnet wird, auch innerhalb dieser geschlossen wird. Nicht-lineare Steuerflüssen erfordern jedoch, das Schließen der inneren Verzweigung hinter das Schließen der äußeren Verzweigung zu legen, also den linearen Steuerfluss zu durchbrechen.

Abbildung 2 illustriert das Problem am Beispiel einer parallelen Komposition.

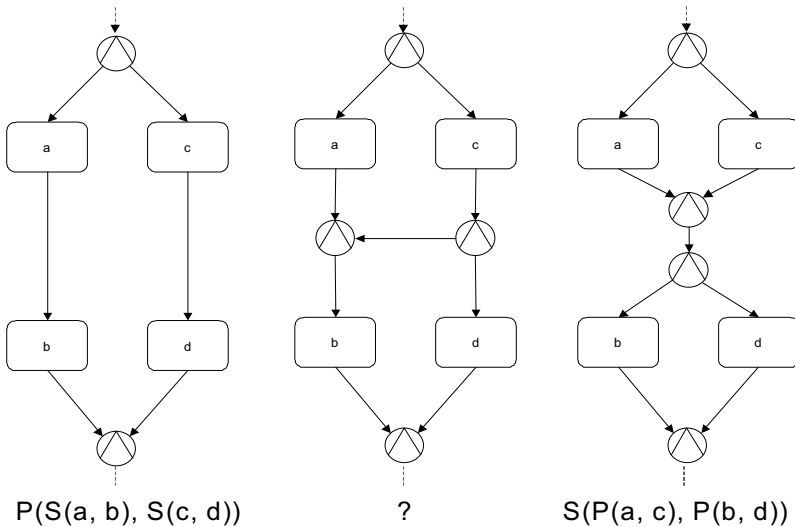


Abbildung 2: Modellierungsproblematik nicht-linearer Steuerflüsse

Für alternative Kompositionen wird in [KHB00] gezeigt, dass durch eine Vervielfachung der im Steuerfluss eingebetteten Blöcke ein dem Ausbruch semantisch entsprechendes Modell auch unter Beibehaltung eines linearen Steuerflusses möglich ist. Entscheidend ist hier die Erweiterung der herkömmlichen Interpretation der Operanden in der Form, dass ein Operand als Referenz interpretiert wird. So können in einem Prozessmodell bspw. beliebig viele Referenzen auf genau eine Aktivität existieren. Da eine alternative Komposition immer genau einen Block auswählt, werden nie mehrere Referenzen der selben Aktivität gleichzeitig aktiviert. Die Modellierung nicht-linearer Steuerflüsse erfolgt, indem zunächst alle bei der vorliegenden Entscheidungsstruktur möglichen Pfade enumeriert werden. Die hiermit verbundene Verschiebung von Entscheidungszeitpunkten wird durch eine anschließende Termersetzung mit $A(S(x_1, \dots, x_n, y_1), \dots, S(x_1, \dots, x_n, y_m)) \rightarrow S(x_1, \dots, x_n, A(y_1, \dots, y_m))$ korrigiert. Die Termersetzungsregel basiert auf der Linksdistributivität der alternativen über die sequentielle Komposition.

Die aufgezeigte Lösung gilt leider nicht für Operatoren, die mehr als einen Pfad gleichzeitig ansteuern. Zur Modellierung nicht-linear verschachtelter, paralleler Kompositionen ist daher eine erneute Erweiterung der Interpretation der eingebetteten Operanden notwendig. Sie erlaubt ebenfalls eine Vervielfältigung der Blöcke sowie Axiome der Links- und Rechtdistribution der parallelen über die sequentielle Komposition.

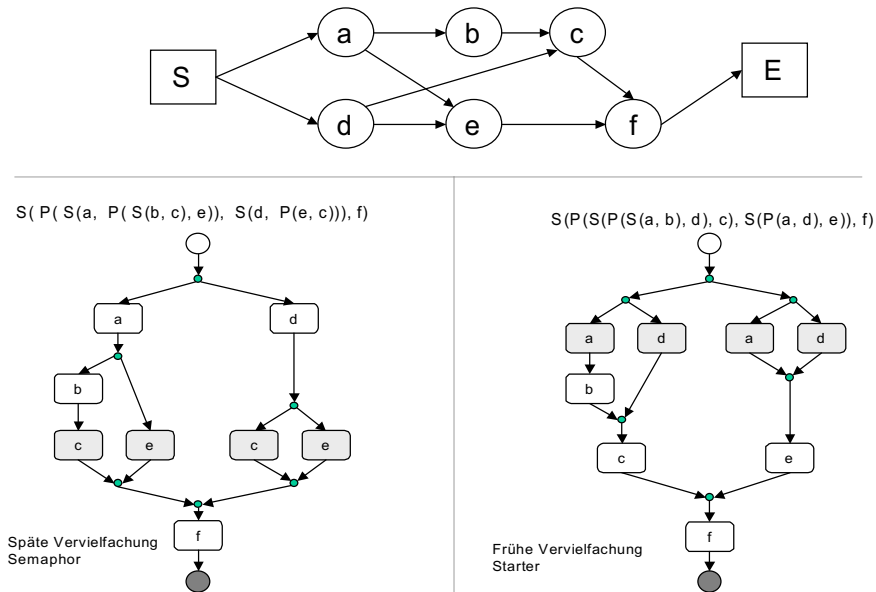


Abbildung 3: Späte und frühe Vervielfachung

Es können zwei Ansätze unterschieden werden, die als späte und frühe Vervielfältigung bezeichnet werden. Beiden ist gemeinsam, dass zwischen Referenz und Instanz ein Element eingefügt wird, das den nebenläufigen Zugriff der Referenz auf die Instanz koordiniert. Beim Ansatz der frühen Vervielfältigung handelt es sich bei dem koordinierenden Element um einen sogenannten Starter. Die Aktivierung der Referenz wird an den Starter weitergeleitet, der beim ersten Aufruf die entsprechende Instanz sofort startet. Alle folgenden Referenzen, die diesen Starter erreichen, verursachen keinen weiteren Start mehr. Wird die Instanz noch ausgeführt, so warten die nebenläufigen Prozesse gemeinsam auf ihre Beendigung. Es findet also eine Synchronisation statt. Erreicht eine Referenz den Starter nachdem die Instanz bereits beendet wurde, so schaltet der Starter durch. Starter können jedem Aktivitätsblock vorgeschaltet werden, da sie beim ersten Erreichen bereits die verbundene Instanz der betreffenden Aktivität starten. Die lineare Modellierung der frühen Vervielfältigung erfolgt, indem nach der Enumeration aller parallelen Sequenzen eine Termersetzung mit $P(S(x_1, y_1, \dots, y_m), \dots, S(x_n, y_1, \dots, y_m)) \rightarrow S(P(x_1, \dots, x_n), y_1, \dots, y_m)$ angewendet wird. Diese Termersetzung basiert auf der Rechtsdistributivität der parallelen über die sequentielle Komposition. Sie kann aufgrund der vorgenommenen Interpretationserweiterung eingeführt werden.

Beim Ansatz der späten Vervielfältigung handelt es sich bei dem koordinierenden Element um einen Semaphor. Die Aktivierung der Referenz wird an den Semaphor weitergeleitet, der erst beim Erreichen aller Referenzen die entsprechende Instanz startet. Das impliziert, dass die korrekte Anzahl der Referenzen dem Semaphor bekannt sein muss. Die lineare Modellierung der späten Vervielfältigung erfolgt, indem nach der Enumeration aller parallelen Sequenzen eine Termersetzung mit $P(S(x_1, \dots, x_n, y_1), \dots, S(x_1, \dots, x_n, y_m)) \rightarrow S(x_1, \dots, x_n, P(y_1, \dots, y_m))$ vorgenommen wird. Diese Termersetzung basiert auf der Linksdistributivität der parallelen über die sequentielle Komposition. Auch sie kann aufgrund der vorgenommenen Interpretationserweiterung eingeführt werden.

In der Abbildung 4 werden beide Ansätze anhand eines Beispiels gegenübergestellt. Sie erscheinen aus der Sicht der Modellierung zunächst äquivalent. Bei der Kombination paralleler und alternativen Kompositionen zur Modellierung nicht-linearer Steuerflüsse führt die frühe Vervielfältigung jedoch zu einer unzulässigen Vorverschiebung der Entscheidungszeitpunkte, die nicht durch eine Termersetzung korrigiert wird. Es wird daher der Ansatz der späten Vervielfältigung eingesetzt. Hierbei ist jedoch zu berücksichtigen, dass zur Bestimmung der Anzahl der Referenzen, die dem Semaphor zugewiesen wird, nur die Referenzen berücksichtigt werden dürfen, die im selben Pfad des Prozessmodells in DNF liegen.

5 Die Prozess-Workbench

Die Prozess-Workbench ist ein Werkzeug zur Modellierung, Analyse und Simulation von linearen Prozessmodellen. Der Anwender kann beliebig komplexe, lineare Prozessmodelle in der Workbench modellieren. Das Werkzeug bietet ihm hierfür in jedem Modellierungsschritt eine Vielzahl von Funktionen, die auf einzelne Aktivitäten genauso gut wie auf Prozessfragmente oder ganze Prozessmodelle angewendet werden können. Hierzu gehören das Ausschneiden, Einfügen, Klonen und Ersetzen von Blöcken in Verbindung mit Zwischenablagen und Schablonen, sowie algebraische Umformungen. Zusammen mit der Verwendung von generischen Blöcken wird sowohl ein Top-down- als auch ein Bottom-up-Modellieren ermöglicht. Die Modellierung defekter Prozesse ist in der Workbench durch die Verwendung des linearen Meta-Prozessmodells jedoch ausgeschlossen.

Die Modellierung eines Prozessmodells beginnt mit einem generischen Block. Dieser ist die Wurzel des Modells. Alle Blöcke lassen sich beliebig in andere Blöcke umwandeln. Werden Operatoren umgewandelt, so bleiben hierbei die in ihnen eingebetteten Blöcke erhalten. Ausgehend von der Wurzel kann durch die Umwandlung generischer Blöcke in Operatoren zunächst top-down die Steuerungslogik des Prozesses modelliert werden. Hierbei brauchen vorerst nur generische Blöcke in den modellierten Steuerfluss eingefügt werden. Unabhängig davon kann eine beliebige Anzahl detaillierter Prozessfragmente modelliert und abgelegt werden. Mit diesen Fragmenten können durch ihr Einsetzen in generische Blöcke auch bottom-up Prozessmodelle erstellt werden. Bei der Modellierung nicht-linearer Steuerflüsse unterstützt die Prozess-Workbench den Modellierer durch die

automatische Umformung alternativer und paralleler Kompositionen und den Einsatz von Prozessschablonen.

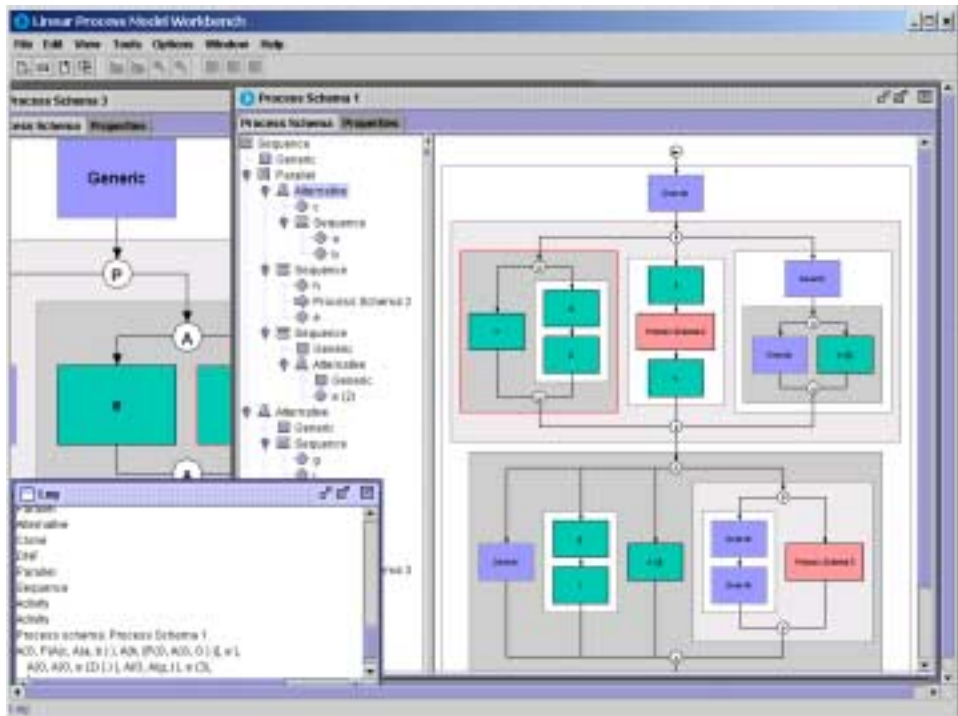


Abbildung 4: Prozess-Workbench

Die Prozess-Workbench ist als Java-Anwendung implementiert. Sie kann somit auf nahezu allen Plattformen eingesetzt werden. Die Workbench bietet dem Anwender eine graphische Benutzeroberfläche, über die alle Funktionen einfach erreichbar sind.

6 Zusammenfassung und Ausblick

Lineare Prozessmodelle besitzen einerseits vorteilhafte Eigenschaften, ihre Praxistauglichkeit war andererseits jedoch durch Defizite eingeschränkt. Dieser Beitrag adressierte die Modellierung nicht-linearer Steuerflüsse und damit das Problem einer eingeschränkten Ausdrucksmächtigkeit, sowie die Bottom-up-Modellierung mit linearen Prozessmodellen als Erweiterung der Ausdrucksfreiheit.

Es wurde gezeigt, dass nicht-lineare Steuerflüsse durch eine Erweiterung der Interpretation der Operanden in Verbindung mit der Vervielfältigung von Blöcken als lineare Prozessmodelle modelliert werden können. Als Lösungsansatz für das Problem der ausschließlichen top-down Modellierung und der mit ihr vom Modellierer abverlangten

hohen Abstraktionsleistung wurden generische Blöcke vorgestellt, durch die in Verbindung mit einem Modellierungswerkzeug sowohl eine Top-down- als auch eine Bottom-up-Modellierung möglich ist.

Ein Schwerpunkt künftiger Forschungsarbeit zu linearen Prozessmodellen wird in der Entwicklung weiterer Operatoren und Schablonen gesehen. Hierdurch können spezialisierte Modelle für einzelne Anwendungsdomänen abgeleitet werden. Mit dieser Erweiterung müssen formale Semantiken und eine entsprechende Anpassung der Algebren einher gehen.

Literatur

- [Aal98] W.M.P. van der Aalst. The Application of Petri Nets to Workflow Management. In The Journal of Circuits, Systems and Computers, 8(1): 21 - 66, 1998.
- [Aal00] W.M.P. van der Aalst. Workflow Verification: Finding Control-Flow Errors using Petri-Net-Based Techniques. In W.M.P. van der Aalst, J. Desel, A. Oberweis (Eds.). Business Process Management – Models, Techniques and Empirical Studies. LNCS 1806. Springer 2000.
- [Bas98] T. Basten: In Terms of Nets: System Design with Petri Nets and Process Algebra. PhD Thesis. Eindhoven University of Technology, Department of Computing Science, Eindhoven, the Netherlands, 1998.
- [BaW90] J.C.M. Baeten, W.P. Weijland: Process Algebra, Cambridge Tracts in Theoretical Computer Science 18, Cambridge University Press 1990.
- [KHB00] B. Kiepuszewski, A.H.M. ter Hofstede, C. Bussler: On Structured Workflow Modelling. In B. Wangler and L. Bergman (Hrsg.): Proceedings Twelfth International Conference on Advanced Information Systems Engineering (CAiSE'2000), volume 1789 of Lecture Notes in Computer Science, pages 431-445, Stockholm, Sweden, June 2000.
- [Rit00] P. Rittgen. Quo vadis EPK in Aris? Ansätze zu syntaktischen Erweiterungen und einer formalen Semantik. In Wirtschaftsinformatik 42, 2000.
- [Rit98] P. Rittgen: Prozeßtheorie der Ablaufplanung, Teubner 1998.
- [Rum99] F. J. Rump: Geschäftsprozessmanagement auf der Basis ereignisgesteuerter Prozeßketten. Teubner 1999.
- [Sch00] G. Schimm: Generic Linear Business Process Modeling. In: Stephen W. Liddle, Heinrich C. Mayer, Bernhard Thalheim (Eds.): Conceptual Modeling for E-Business and the Web. Proceedings of the ER 2000 Workshop on Conceptual Approaches for E-Business and The World Wide Web and Conceptual Modeling. Salt Lake City, Utah, USA, 2000. LNCS 1921.