Korrektheitsaussagen über Prozeßmodelle anhand von Transaktionshüllen

Thomas Herstel

Otto-von-Guericke-Universität Magdeburg
Institut für Technische und Betriebliche Informationssysteme
Postfach 4120, D-39016 Magdeburg
herstel@iti.cs.uni-magdeburg.de

Zusammenfassung

Ein Ansatz zur wettbewerbsbedingten Verkürzung der für die Produktentwicklung benötigten Zeitspanne ist die rechnergestützte Steuerung der Ausführung des Entwicklungsprozesses. Die Ausführung erfolgt nach Vorgabe eines Prozeßmodells, daß Kontroll- und Datenfluß des zu unterstützenden Prozesses abbildet. Demgegenüber existieren erweiterte Transaktionskonzepte, die eine ähnliche Funktionalität anbieten. Das Konzept der Transaktionshülle bietet unter anderem für diese Transaktionsmodelle einen einheitlichen formalen Rahmen der Beschreibung an und erlaubt zusätzlich eine Überprüfung der Spezifikation auf Korrektheit. Durch Abbildung eines Prozeßmodells auf Transaktionshüllen können Aussagen über die Korrektheit des Prozeßmodells hinsichtlich des Kontroll- und Datenflusses getroffen werden.

1 Einleitung

Unternehmen zum Beispiel im Bereich des Maschinen- und Anlagenbaus sind heute einem immer stärkeren Wettbewerbsdruck ausgesetzt. Um diesem Wettbewerbsdruck beispielsweise auf dem Gebiet der Produktentwicklung entgegenwirken zu können, werden heutzutage zunehmend informationstechnische Konzepte eingesetzt. Neben der Unterstützung der entwickelnden Ingenieure durch entsprechende Softwaresysteme können die auszuführenden Aktivitäten rechnerunterstützt koordiniert werden. Dazu ist die Abbildung des Prozesses auf sogenannte Prozeßmodelle notwendig.

1.1 Prozeßmodelle

Die Steuerung des Prozesses erfolgt dann durch Ausführung des Prozeßmodells. Prozeßmodelle können zum einen genutzt werden, um einen bestehenden Prozeß hinsichtlich seiner Effizienz zu bewerten, um gegebenenfalls Optimierungen, zum Beispiel durch Erhöhen des Anteils nebenläufig ausführbarer Aktivitäten im Prozeß, vornehmen zu können. Aus informationstechnischer Sicht steht jedoch der Aspekt der möglichen (rechnergestützten) Ausführung und Kontrolle eines Prozesses im Vordergrund.

Für die Erstellung von Prozeßmodellen gibt es verschiedene Paradigmen [LC91]. Dazu zählen Ansätze aus dem Bereich der künstlichen Intelligenz, aktiver Datenbanken, der Prozeßprogrammierung oder graphische/netzbasierte Ansätze. Letztere eignen sich besonders durch ihre höhere Anschaulichkeit und Visualisierbarkeit des Zustands des Prozesses während der Ausführung. Beispiele für konkrete Modelle dieses Ansatzes sind Petrinetze [Jab95], Funsoft-Netze ([EG91, Gru91] und Ereignisgesteuerte Prozeßketten [Sta99]. Neben der graphischen Darstellbarkeit eines Prozeßmodells ist wichtig, daß sowohl der Kontroll- als auch der Datenfluß eines Prozesses abgebildet werden können, die verwendete Methode selbst formal fundiert ist sowie alle wesentlichen Grundkonstrukte der Prozeßmodellierung unterstützt werden. Der Kontrollfluss eines Prozeßmodells wird aus diesen Grundkonstrukten zusammengesetzt. Als wesentliche Kontrollflußgrundkonstrukte gelten die Ausführung in Sequenz, Parallelität (AND-Split), die bedingte (XOR-Split) und die allgemeine (OR-Split) Verzweigung sowie die entsprechenden Zusammenführungen des Kontrollflusses (Joins) und Iterationen [Jab95]. Weitere Anforderungen an Prozeßmodelle sind beispielsweise die Unterstützung eines Rollenkonzepts oder eines Zeitmodells.

Von den genannten Prozeßmodellierungsmethoden der Petrinetze, Funsoft-Netze und Ereignisgesteuerten Prozeßketten bieten Funsoft-Netze die umfassendste Funktionalität. Beispielsweise erlauben sie als einzige die Darstellung des Datenflusses des Prozesses. Charakteristisch für Funsoft-Netzen, die eine höhere Form von Petrinetzen sind, ist die vergleichsweise kompakte Modellierung der Kontrollflußgrundkonstrukte durch verschiedene Ausgangschaltverhalten und die Verwendung typisierter Stellen für den Datenaustausch.

1.2 Transaktionshüllen

Bedingt durch das große Interesse an der rechenergestützen Steuerung von Prozeßabläufen wurden auch im Datenbankbereich Techniken untersucht und eingeführt, die dieses Anwendungsfeld abzudecken versuchen. Die innerhalb eines Prozesses auszuführnden Arbeitsschritte werden als Transaktionen aufgefaßt, die es zu koordinieren gilt. Gerade für den Bereich der Nicht-Standardanwendungen – wie CAD/CAM, Softwareentweicklungsumgebungen und den Workflowbereich – wurden zur Vermeidung von Problemen mit dem traditionellen ACID-Prinzip erweiterte Transaktionsmodelle entwickelt. Transaktionshüllen ([Sch99, STS98]) bieten einen einheitlichen formalen Rahmen für die Beschreibung von Transaktionen und den zwischen ihnen geltenden Abhängigkeiten. Auszuführende Aktivitäten werden als Transaktionen aufgefaßt, die miteinander in Beziehung stehen. Transaktionshüllen erlauben sowohl eine hierarchische Strukturierung von Transaktionen über Vater-Kind-Beziehungen als auch die Spezifikation sogenannter Terminierungs-, Ausführungs- und Objektsichtbarkeitsabhängigkeiten. Terminierungsabhängigkeiten geben an, welche kKombination von COMMIT und ABORT zwischen den einzelnen Transaktionen erlaubt beziehungsweise verboten sein sollen. Auf diese lassen sich lassen sich zueinander exklusive Transaktionen oder einander bedingende Transaktionen spezifizieren. Ausführungsabhängigkeiten beschreiben, ob Transaktionen sequentiell nacheinander, oder parallel ausgeführt werden können, wobei zusätzlich zwischen einschließender und überlappender Parallelität unterschieden wird. Die Objektsichtbarkeitsabhängigkeiten legen zusätzlich fest, in welcher Art und Weise der Datenaustausch zwischen Transaktionen erfolgen soll.

Der Vorteil der Spezifikation von Transaktionshüllen ist ihre Überprüfbarkeit auf Konsistenz. Dazu werden nach einem Regelwerk zunächst aus den spezifizierten Abhängigkeiten alle möglichen Abhängigkeiten hergeleitet, die daraus transitiv folgen. Unter den so erhaltenen Abhängigkeiten kann geprüft werden, ob sich Widersprüche ergeben, beispielsweise wenn eine Transaktion auf jeden Fall abgebrochen werden muß, das heißt nie zur Ausführung gelangt.

2 Abbildung von Funsoft-Netzen auf Transaktionshüllen

Für Funsoft-Netze existiert eine solche direkte Prüfung der Konsistenz nicht. Allerdings ist es möglich, das Funsoft-Netz auf andere einfacherere Petrinetze zurückzuführen und dort entsprechende Algorithmen (zum Beispiel aus der linearen Algebra) anzuwenden. Die Anwendung solcher Algorithmen erscheint im allgemeinen jedoch sehr aufwendig und ist daher in gewissem Sinne nicht praktikabel. Um die vergleichsweise einfache Überprüfung der Transaktionshüllen auf Konsistenz auch für das Prozeßmodell der Funsoft-Netze einsetzen zu können, ist es notwendig, die Funsoft-Netze auf Transaktionshüllen abzubilden. Ein solcher Abbildungsalgorithmus wurde in [Her99] hergeleitet. Zunächst ist es notwendig, einen Abgleich der Semantik beider Modellierungsmethoden vorzunehmen. Insbesondere ist dafür die Einführung der Konvention notwendig, daß auch Prozesse, die mit Funsoft-Netzen modelliert wurden, abbrechen können, obwohl eine solche Möglichkeit dort ursprünglich nicht vorgesehen ist. Ein erster und intuitiver Ansatz zur Abbildung der Funsoft-Netze auf Transaktionshüllen bestand darin, diese als Zusammensetzung von Kontrollflußgrundkonstrukten aufzufassen und diese Kontrollflußgrundkonstrukte unabhängig voneinander in ihre Entsprechung in Transaktionshüllen zu überführen. Jedem Kontrollflußgrundkonstrukt des Prozeßmodells würde eine entsprechende Beschreibung in der Transaktionshülle zugeordnet. Dieser Ansatz erzeugt jedoch unter Umständen für korrekte Funsoft-Netze Transaktionshüllen, deren Spezifikation inkonsistent ist. Eine Ursache dafür sind Iterationen, die in Transaktionshüllen nicht unterstützt werden und deren Abbildung zu zyklischen Abhängigkeiten. Zum Beispiel kann eine Spezifikation fordern, daß eine Transaktion erst dann starten darf, wenn sie bereits beendet worden ist. Aber auch Funsoft-Netze, die frei von Iterationen sind, können fehlerhaft abgebildet werden, weil für ihre korrekte Abbildung auf Transaktionshüllen Wissen über das Zusammenspiel der einzelnen Grundkonstrukte benötigt wird, daß bei der separaten unabhängigen Abbildung verloren geht.

Da Transaktionshüllen das Grundkonstrukt der Iteration nicht unterstützen, müssen Funsoft-Netze, die Iterationen enthalten, so abgebildet werden, als ob sie keine Iterationen ausführen würden. Die Erkennung und Behandlung der Iterationen beruht auf der Erkennung der Zyklen im Graphen der Funsoft-Netze. Diese kann mit Hilfe eines Pfadansatzes erfolgen. Iterationen können semantisch in Transaktionshüllen durch Einführung eines speziellen Konstrukts, einer sogenannten „intelligenten Vatertransaktion“ ausgedrückt werden. Diese kapselt die Ausführung der zu iterierenden Transaktionen durch die gesteuerte Ausführung zweier Teile – ein Teil für einmalige Ausführung (ohne Rücksprung) an den sich ein zweiter Teil (gesamte Iteration) anschließen kann, der beliebig oft wiederholt wird. Durch eine derartige Ersetzung von Iterationen lassen sich die dadurch verursachten Probleme beheben. Jedoch ist für die korrekte Abbildung eine neue Herleitung der Abhängigkeiten aus Funsoft-Netzen notwendig. Dazu werden für Funsoft-Netze mögliche Ausführungszustände bestimmt und aus der Kombination der möglichen Ausführungszustände lassen sich die Terminierungsabhängigkeiten neu herleiten. Aus der systematischen Untersuchung von Kombinationen von Grundkonstrukten in Funsoft-Netzen und Abbildung auf Terminierungsabhängigkeiten konnte ein Regelwerk zur allgemeinen Erzeugung von Terminierungsabhängigkeiten abgeleitet werden. Die Generierung der anderen Abhängigkeiten (Ausführungs- und Objektsichtbarkeitsabhängigkeiten) ergibt sich in jedem Falle aus den Sequenzen von Aktivitäten in Funsoft-Netzen und beschränkt sich in der Transaktionshülle ausschließlich auf die Festlegung der sequentiellen Reihenfolge der Transaktion beziehungsweise der Übergabe von Daten mit dem Ende der Transaktion.

3 Korrektheitsaussagen

Ausgehend von der Möglichkeit der Abbildung konnte untersucht werden, inwiefern nicht korrekte Funsoft-Netze zu einer inkonsistenten Spezifikation in Transaktionshüllen führen. Funsoft-Netze wurden im Rahmen dieser Arbeit als dann nicht korrekt betrachtet, wenn die verwendeten Grundkonstrukte nicht typgleich kombiniert (dem Blockansatz folgend) wurden. Die erhoffte Vereinfachung der Überprüfung einer Spezifikation eines Funsoft-Netzes auf Ebene der Transaktionshüllen ließ sich in einem Fall feststellen: Die Abbildung einer Kombination aus XOR-Split/AND-Join führt zu einer (erwünschten) Inkonsistenz in der Transaktionshülle. Die beiden anderen möglichen Kombinationen von typfremden Split- und Join-Konstrukten führen nicht zu widersprüchlichen Abhängigkeiten und können erst durch eine pfadbasierte Untersuchung der Transaktionshülle erkannt werden.

Korrektheitsaussagen anhand des Datenflusses, wie er in Funsoft-Netzen modelliert wird, sind möglich. Allerdings treten nur dann widersprüchliche Objektsichtbarkeitsabhängigkeiten in der Transaktionshülle auf, wenn auch widersprüchliche sequentielle Abhängigkeiten erzeugt werden. Beides ist nur dann möglich, wenn das Funsoft-Netz einen Zyklus enthält. Da für die Abbildung des Prozeßmodells der Funsoft-Netze auf Transaktionshüllen jedoch der Zyklus eliminiert wird, können mit den bestehenden Objektsichtbarkeitsabhängigkeiten, die sich aus der Verwendung der Modellierung der Datenübergabe mit Funsoft-Netzen ergeben, keine Inkonsistenzen auftreten.

Doch weder Funsoft-Netze noch die bestehenden Objektsichtbarkeitsabhängigkeiten in Transaktionshüllen bilden Datenflußbeziehungen, wie sie für den Bereich der Produktentwicklung üblich sind, adäquat ab. Bedingt durch die Schaltsemantik von Petrinetzen kann die Datenübergabe wie sie in Funsoft-Netzen modelliert ist, ausschließlich sequentiell erfolgen. Für die Produktentwicklung ist jedoch eine gemeinsame Nutzung der Datenobjekte und die Weitergabe von Datenobjekten auch während der Ausführung einer Aktivität typisch. Daher muß die Modellierung der Weitergabe von Daten in einem Modell erfolgen, daß die nebenläufige Benutzung der Daten zuläßt, zum Beispiel Datenflußdiagramme. In Transaktionshüllen ist die Einführung einer neuen Objektsichtbarkeitsabhängigkeit notwendig, die im Gegensatz zu den bekannten Objektsichtbarkeitsabhängigkeiten eine Datenübergabe unabhängig vom Beginn der empfangenden beziehungsweise Ende der sendenden Transaktion ermöglicht.

Um die Zulässigkeit eines Zugriffs auf ein Datenobjekt in einer Systemumgebung in der Produktentwicklung feststellen zu können werden die Datenobjekte mit einem Status gekennzeichnet, der anzeigt, in welchem Zustand sich das Objekt zum Zeitpunkt des Zugriffs befindet. Die möglichen Zustände sowie die möglichen Wechsel von einem Zustand in einen anderen können mit Hilfe von Zustandsdiagrammen modelliert werden. Um in einer Transaktionshülle Aussagen darüber treffen zu können, ob eine unzulässige Reihenfolge des Zugriffs (Inkonsistenz) vorliegt oder nicht, ist die Erweiterung der Objektsichtbarkeitsabhängigkeit um die Information, welches Objekt ausgetauscht wurde und mit welchem Status es dazu versehen wurde, notwendig. Eine Objektsichtbarkeitsabhängigkeit die Objekt, Status sowie sendende und empfangende Transaktionen zusammen mit einer unabhängigen Datenübermittlung vereint, konnte angegeben werden.

Die Möglichkeit der Überprüfung der Konsistenz der Abfolge einer Datenübermittlung ergibt sich aus dem Zustandsdiagramm. Es konnte ein Algorithmus angegeben werden, der ein beliebiges Zustandsdiagramm in überprüfbare Bedingungen im Rahmen des ACTA-Formalismus erzeugt. Dies ermöglicht auch die Konsistenzprüfung von Datenflußbeziehungen auf Ebene der Transaktionshüllen.

4 Zusammenfassung

Um Arbeitsabläufe rechnergestützt steuern zu können, werden verschiedene Ansätze der Prozeßmodellierung verfolgt. Auf dem Gebiet der graphischen/netzbasierten Methoden sind Funsoft-Netze im Vergleich zu einfachen Petrinetzen und Ereignisgesteuerten Prozeßketten ausdrucksmächtiger, da sie beispielsweise die gleichzeitige Abbildung von Kontroll- und Datenfluß ermöglichen. Auf der anderen Seite bestehen im Bereich der Datenbanken erweiterte Transaktionskonzepte, die ebenfalls eine Modellierung von Prozessen zulassen. Diese koennen mittels eines einheitlichen und formalen Rahmens, den Transaktionshüllen, beschrieben werden, für die im Gegensatz zu Funsoft-Netzen eine direkte Überprüfbarkeit der Spezifikation auf Konsistenz möglich ist. Um diese Möglichkeit der Überprüfung auch auf graphische/netzbaierte Prozeßmodelle anwenden zu können, ist eine Abbildung auf die Transaktionshüllen notwendig. Inkonsistente Kontrollflüsse lassen sich so z.T. leichter erkennen. Eine Überprüfung des Datenflusses ist nach Erweiterung der Transaktionshüllen ebenso möglich.

Literatur

[EG91] Wolfgang Emmerich and Volker Gruhn. FUNSOFT Nets: A Petri-Net based Software Process Modelling Language. In *Proceedings of the 6^{th} International Workshop on Software Specification and Design*, pages 175–184, Como, Italy, September 1991. IEEE Computer Society Press.

[Gru91] Volker Gruhn. Validation and Verification of Software Process Models. Dissertation, Technical Report 394/91, University of Dortmund, June 1991.

[Her99] T. Herstel. Entwicklung einer Abbildungsmethode zwischen einem Prozeß- und einem Transaktionsmodell zur Ableitung von Korrektheitsaussagen für Datenflußabhängigkeiten im Engineering-Bereich. Diplomarbeit, Universität Magdeburg, Fakultät für Informatik, 1999.

[Jab95] S. Jablonski. *Workflow-Management-Systeme: Motivation, Modellierung, Architektur*, volume 9 of *Thomson's Aktuelle Tutorien*. International Thomson Publishing, Bonn, 1995.

[LC91] Chunnian Liu and Reidar Conradi. Process Modelling Paradigms: An Evaluation. In *First European Workshop on Software Process Modeling, Milano, Italy*, pages 39–52, May 1991.

[Sch99] K. Schwarz. *Das Konzept der Transaktionshüllen.* PhD thesis, Universität Magdeburg, 1999.

[Sta99] Josef Staud. *Geschäftsprozeßanalyse mit Ereignisgesteuerten Prozeßketten: Grundlagen des Business Process Reengineering für SAP R/3 und andere betriebswirtschaftliche Standardsoftware.* Springer-Verlag, Berlin, 1999.

[STS98] K. Schwarz, C. Türker, and G. Saake. Transitive Dependencies in Transaction Closures. In B. Eaglestone, B. C. Desai, and J. Shao, editors, *Proceedings of the 1998 International Database Engineering and Application Symposium (IDEAS'98), July 8–10, 1998, Cardiff, Wales, UK*, pages 34–43, Los Alamitos, CA, 1998. IEEE Computer Science Press.