Prozessumstrukturierung unter Berücksichtigung von Nachrichteninhalten

Thomas S. Heinze[1], Wolfram Amme[1], Simon Moser[2]

[1] Friedrich-Schiller-Universität Jena
{T.Heinze,Wolfram.Amme}@uni-jena.de
[2] IBM Entwicklungslabor Böblingen
smoser@de.ibm.com

1 Einführung und Motivation

Bei der informationstechnischen Umsetzung von Geschäftsprozessen mit Hilfe von verteilten, service-orientierten Architekturen existieren mehrere sowohl wissenschaftlich als auch industriell relevante Fragestellungen. Neben grundsätzlichen Fragen zur Korrektheit und Kompatibilität verteilter Prozesse spielt auch das Auffinden geeigneter Dienste zur Dienstkomposition eine wichtige Rolle. Grundlage einer solchen Dienstsuche bilden öffentliche Schnittstellenbeschreibungen, die zumeist aus einer Liste der implementierten Operationen der Dienste bestehen. Um Fehler bei der Dienstkomposition zu vermeiden, sollten insbesondere bei zustandsbehafteten Diensten (wie zum Beispiel Geschäftsprozessen) zusätzlich Informationen zu der Abfolge der Operationen vorhanden sein. Ein formaler Ansatz zur Modellierung und Analyse der durch einen (verteilten) Geschäftsprozess realisierten Abfolge von Operationen, in Form gesendeter und empfangener Nachrichten, wird in [6, 7] beschrieben. Darin wird das Prozessverhalten aus Sicht eines möglichen Partners wiedergegeben und auf diese Weise eine Art *Bedienungsanleitung* für den Prozess bereitgestellt. Diese kann dann zur Schnittstellenbeschreibung genutzt werden und so die Dienstsuche unterstützen [7].

Abbildung 1 zeigt das Fragment eines Geschäftsprozesses, in diesem Fall der Sprache WS-BPEL [8], zusammen mit der das zugehörige Verhalten beschreibenden Bedienungsanleitung. Die dargestellte Aktivität `BiddingSequence` setzt ein Veräußerungsverfahren um, bei dem in einer Schleife (`RepeatUntil`) fortlaufend eine Ausschreibung veröffentlicht wird (`BidRequest`), zu der ein Gebot abgegeben werden kann (Nachricht `Bid`). Die Schleife wird verlassen und das Verfahren mit einer Mitteilung beendet (`BidClosure`), falls ein Gebot den vordefinierten Mindestverkaufswert übersteigt. Darüber hinaus kann das Verfahren auch durch den Veräußerer vorzeitig abgebrochen werden (Nachricht `Abort`). Die Ausführung der Schleife wird dabei durch drei Variablen kontrolliert (`$break or $currentBid` $>$ `$threshold`). Zum einen enthalten `$currentBid` und `$threshold` Informationen zu dem zuletzt abgegebenen Gebot und dem Mindestverkaufswert (`1000`). Andererseits wird der vorzeitige Abbruch des Verfahrens durch `$break` gesteuert. Wie ebenfalls in Abbildung 1 dargestellt, ist die Bedienungsanleitung eines Geschäftsprozesses ein Automat [7]. In diesem

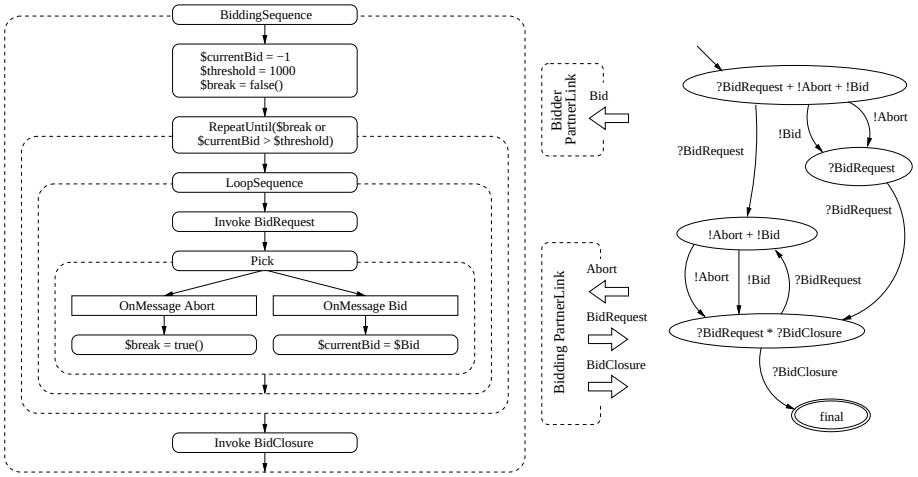

Abb. 1. `BiddingSequence` (links) und zugehörige Bedienungsanleitung (rechts)

repräsentiert ein Pfad eine mögliche Ausprägung des von außen sichtbaren Prozessverhaltens, wobei die Kanten des Automaten den gesendeten und empfangenen Nachrichten aus Sicht des jeweiligen Prozesspartners entsprechen. Zusätzlich finden sich Integritätsbedingungen an die möglichen Partnerprozesse in Form von Zustandsannotationen.[1] Wie leicht zu erkennen ist, beschreibt die angegebene Bedienungsanleitung jedoch nicht alle denkbaren Partner der Aktivität `BiddingSequence`. So sind beispielsweise Partnerprozesse, die zunächst mehrere Gebote (Nachricht `Bid`) senden, und erst im Anschluss die zugehörigen Ausschreibungen (`BidRequest`) empfangen, nicht mit erfasst. Auch ist das Ende der Veräußerung nach einem den Mindestverkaufswert übersteigenden Gebot oder einem vorzeitigen Abbruch (Nachricht `Abort`) nicht explizit wiedergegeben.

Die Begründung dafür ist in dem zur Ableitung der Bedienungsanleitung angewandten Verfahren zu suchen. Dieses beruht auf einer petrinetzbasierten Modellierung der Geschäftsprozesse, in der Prozessdaten zugunsten einer durchführbaren Analyse nicht wiedergegeben werden [5, 6]. Insbesondere wird der bedingte Kontrollfluss auf nichtdeterministische Konflikte abgebildet, und damit eine Abstraktion verwendet, die zu der skizzierten Ungenauigkeit in der abgeleiteten Bedienungsanleitung führen kann. In [4] haben wir eine *Umstrukturierungsmethode* vorgestellt, mit der sich bestimmte Arten des bedingten Kontrollflusses so transformieren lassen, dass Daten- in Kontrollabhängigkeiten umgewandelt werden. Im Ergebnis können die Datenabhängigkeiten, namentlich die Verzweigungs- oder Schleifenbedingungen, entfernt und somit die Anzahl der benötigten nichtdeterministischen Konflikte reduziert werden. Die Klasse der in [4] betrachteten Schleifen und Verzweigungen ist dabei durch Bedingungen charakterisiert, die zur Prozesslaufzeit nur auf durch Konstanten definierte Variablen zugreifen.

[1] Die durch Annotationen ausgeschlossenen Pfade der Bedienungsanleitung, dass heißt Pfade die zu fehlerhaftem Verhalten führen können, sind nicht mit abgebildet.

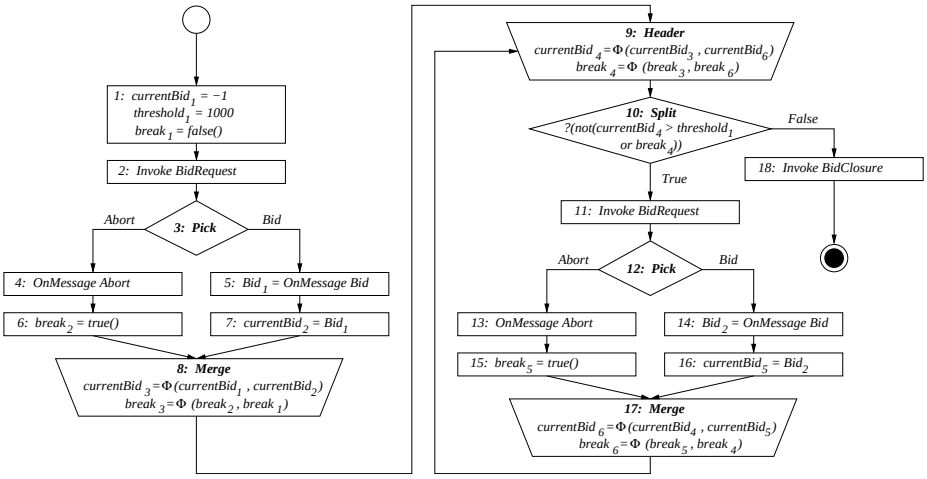

Abb. 2. Erweiterter Workflow-Graph für `BiddingSequence`

Im Folgenden soll ein Ansatz zur Erweiterung der Umstrukturierungsmethode am Beispiel `BiddingSequence` beschrieben werden, mit der sich auch solche Verzweigungs- und Schleifenbedingungen eliminieren lassen, deren Variablen zusätzlich zu Konstanten durch eingehende Nachrichten definiert sind.

2 Prozessumstrukturierung

Die Umstrukturierungsmethode basiert auf einer Prozessrepräsentation durch *erweiterte Workflow-Graphen* und ist damit grundsätzlich auf Prozesse aller Spezifikationssprachen für (strukturierte) Geschäftsprozesse anwendbar, zu denen eine Abbildung auf Workflow-Graphen existiert (außer für WS-BPEL existiert ein Abbildungsverfahren [1] für BPMN [9]). Erweiterte Workflow-Graphen erlauben zusätzlich zu der durch herkömmliche Workflow-Graphen unterstützten Abbildung des Kontrollflusses auch die Wiedergabe der Prozessdaten. Insbesondere lassen sich die während der Umstrukturierung benötigten Datenabhängigkeiten innerhalb dieses Repräsentationsformats auf einfache Weise identifizieren. In Abbildung 2 ist der erweiterte Workflow-Graph für das Beispiel `BiddingSequence` abgebildet.[2] Darin repräsentieren Knoten die Aktivitäten und Kanten verbinden diese entsprechend dem Kontrollfluss. Spezielle Knoten (*Pick*, *Merge*, *Split*, *Header*) dienen der Aufspaltung und Vereinigung des Kontrollflusses im Fall der ereignisgesteuerten Verzweigung (`Pick`) und der Schleife des Prozessfragments. Wie bereits erwähnt, erfolgt die Repräsentation der Prozessdaten derart, dass eine einfache Analyse der Datenabhängigkeiten möglich ist. Zu diesem Zweck wur-

[2] Aus technischen Gründen wurde die eigentlich fußgesteuerte Schleife (`RepeatUntil`) in `BiddingSequence` durch Herausziehen der ersten Iteration und Negierung der Schleifenbedingung in eine äquivalente kopfgesteuerte Schleife umgewandelt.

den die Variablen so umbenannt, das jede statische Variablendefinition einen eigenen Namen erhält (so $break_1, break_2, \ldots, break_6$ für `$break`). Weiterhin wurde eine Φ-Funktion zur Zusammenfassung konkurrierender Definitionen eingefügt, falls mehrere Definitionen einer Variablen auf verschiedenen Pfaden des Kontrollflusses zusammentreffen (vergleiche $break_4 = \Phi(break_3,\ break_6)$ in *Header*). Der Wert einer solchen Funktion entspricht gerade der Definition, gegeben als Operand, die zur Laufzeit ausgeführt wurde.

Ausgehend von dieser Prozessrepräsentation lassen sich die Datenabhängigkeiten des bedingten Kontrollflusses analysieren. Für `BiddingSequence` fällt dabei auf, dass die Bedingung der darin enthaltenen Schleife nur auf Variablen zugreift, die entweder allein durch Konstanten (`$break`, `$threshold`) oder zusätzlich durch Nachrichten (`$currentBid`) definiert sind. Im ersteren Fall wird von *statisch quasi-konstanten Variablen* gesprochen, und für Bedingungen die nur auf diese Art von Variablen zugreifen von *statisch quasi-konstanten Bedingungen*. Der Wert einer solchen Bedingung hängt nur vom zur Prozesslaufzeit gewählten Kontrollflusspfad ab, da jeder Pfad zur Bedingung einer Belegung der darin vorkommenden Variablen mit Konstanten entspricht. Daher kann die Bedingung durch Einfügen geeigneter Kontrollabhängigkeiten ersetzt werden.

Die Umstrukturierungsmethode zur Erzeugung der Kontrollabhängigkeiten erfolgt in zwei Schritten. Zunächst wird eine Schleife oder Verzweigung mit statisch quasi-konstanter Bedingung in eine *Normalform* überführt. Diese zeichnet sich dadurch aus, das alle (statischen) Pfade des Kontrollflusses, die für eine Bedingungsvariable unterschiedliche Werte definieren, aufgetrennt sind. Zu diesem Zweck werden, mit Ausnahme des Kopfs einer Schleife, alle Knoten in denen unterschiedliche Definitionen aufeinandertreffen nacheinander aufgelöst. Im Anschluss kann für eine Verzweigung die Verzweigungsbedingung auf allen Pfaden ausgewertet und durch unbedingte Sprünge zu den Verzweigungszielen ersetzt werden. Für eine Schleife laufen nun hingegen alle Definitionen für Bedingungsvariablen im Schleifenkopf zusammen. Um auch diese aufzutrennen, wird die Schleife im folgenden Schritt durch mehrere *Schleifeninstanzen* ersetzt. Jede *Instanz* repräsentiert dabei die Ausführung des Schleifenrumpfs für eine mögliche Belegung der Bedingungsvariablen mit Konstanten. Demnach können die Schleifenbedingungen in den Instanzen ebenfalls ausgewertet, und durch unbedingte Sprünge zum Austrittsknoten der Schleife oder zu weiteren Instanzen ersetzt werden. Eine ausführliche Beschreibung der Methode ist in [4] zu finden.

3 Erweiterte Umstrukturierung

Nachstehend wird die Erweiterung der Umstrukturierungsmethode beschrieben. Diese soll auch die Elimination von Bedingungen erlauben, in denen Variablen durch eingehende Nachrichten definiert sind. Die Bedingung der Schleife in `BiddingSequence` enthält, neben den statisch quasi-konstanten Variablen `$break` und `$threshold`, auch eine solche Variable (`$currentBid`). Diese Art von Variablen, und Bedingungen die nur auf diese Variablen zugreifen, werden als *dynamisch quasi-konstant* bezeichnet. Um die Umstrukturierungsmethode

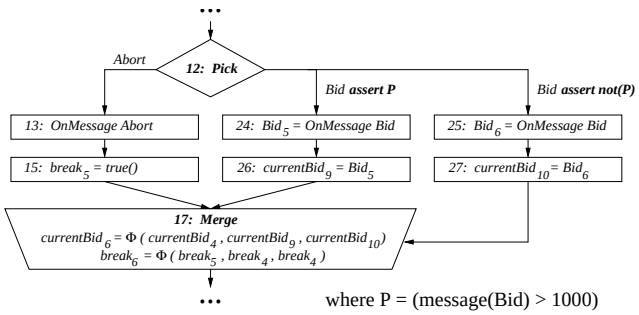

Abb. 3. Verfeinerte Abbildung der die Variable `$currentBid` definierenden Nachricht

auch auf Schleifen und Verzweigungen mit derartigen Bedingungen anwenden zu können, wird ein abstrakterer Instanzenbegriff benötigt. Der bisher verwendete Begriff beschreibt die Ausführung des Rumpfs einer Schleife bezüglich einer Belegung der Bedingungsvariablen mit Konstanten. Im Folgenden verstehen wir unter einer Schleifeninstanz die Ausführung des Schleifenrumpfs bezüglich einer allgemeinen *Zusicherung* für den Zustandsraum der Bedingungsvariablen. Zweifelsohne ist die erstere Begriffsbildung ein Spezialfall der letzteren.

Die Zusicherungen werden dabei aus der betrachteten Bedingung abgeleitet. Dazu wird diese zunächst in eine Klauselform transformiert, aus der die Grundprädikate der Bedingung bestimmt werden können, so ($currentBid_4 > threshold_1$) und ($break_4$) für die Bedingung in `BiddingSequence`. Zwei Typen von Prädikaten müssen unterschieden werden. Zum einen können Prädikate nur auf statisch quasi-konstante Variablen zugreifen ($break_4$). In diesen Fällen werden, wie bisher, Belegungen der Variablen mit Konstanten als Zusicherungen in den Instanzen verwendet. Davon zu trennen sind Prädikate, in denen auch dynamisch quasi-konstante Variablen vorkommen ($currentBid_4 > threshold_1$), da nun das Prädikat auch vom Inhalt eingehender Nachrichten abhängig ist. Um die dadurch gegebenen Datenabhängigkeiten mittels Kontrollabhängigkeiten repräsentieren zu können, werden Nachrichten nicht mehr nur als Ereignisse interpretiert, sondern zusätzlich Klassen möglicher Nachrichteninhalte unterschieden.[3] Dies erlaubt die Definition einer Variablen durch eine Nachricht mit Zusicherungen für die möglichen Werte der Nachricht zu verfeinern. Im erweiterten Workflow-Graphen werden dann die Prädikate mit dynamisch quasi-konstanten Variablen auf Zusicherungen für die diese Variablen definierenden Nachrichten zurückgeführt. In Abbildung 3 ist das Ergebnis für die Definition der Variablen `$currentBid` innerhalb der Schleife aus `BiddingSequence` angegeben. Wie zu erkennen, ist die ursprüngliche Definition der Variablen (Knoten 16 in Abbildung 2) dupliziert und durch zwei komplementäre Zusicherungen an die jeweils definierende Nachricht `Bid` ergänzt worden ($assert\ P$, $assert\ not(P)$). Die Zu-

[3] Die explizite Repräsentation von Nachrichteninhalten wurde bereits in [5] vorgeschlagen, basierend auf der Entfaltung höherer Petrinetze. Dies erfordert jedoch das Vorliegen von Bedingungen und Nachrichten mit endlichen Datenbereichen.

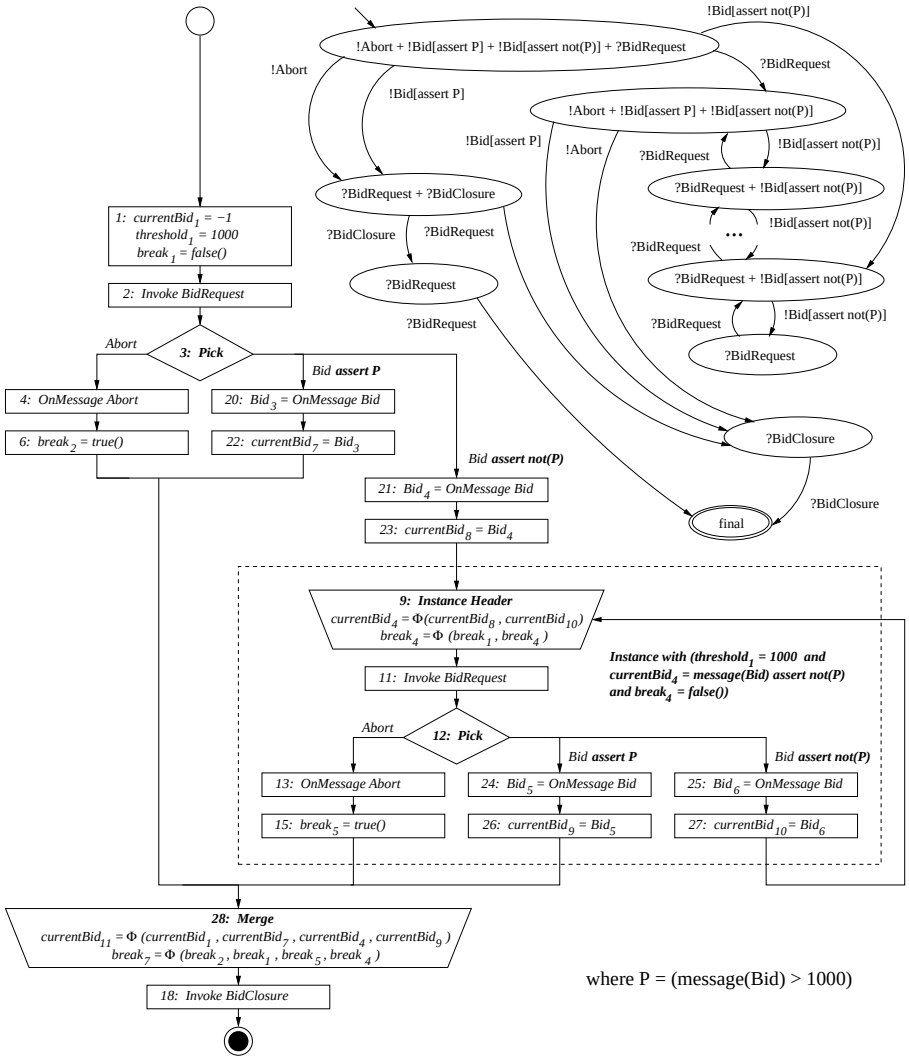

Abb. 4. Umstrukturierter Workflow-Graph und zugehörige Bedienungsanleitung

sicherungen entsprechen dabei der positiven und negativen Variante des die Variable `$currentBid` enthaltenen Prädikats der Schleifenbedingung, in denen die Variablen durch ihre Definitionen ersetzt wurden ($P = message(Bid) > 1000$).

Während der Umstrukturierung können die damit eingeführten Zusicherungen für relevante Nachrichten zur Beschreibung der Schleifeninstanzen genutzt werden. Das Resultat der so erweiterten Umstrukturierung ist für die Schleife aus `BiddingSequence` in Abbildung 4 dargestellt. Darin wurde lediglich eine einzige Instanz mit den Zusicherungen $break_4 = false()$, $threshold_1 = 1000$ und $currentBid_4 = message(Bid)\ assert\ not(P)$ erzeugt, da die Schleifenbe-

dingung nur für diese Zusicherungen erfüllt ist. Abbildung 4 zeigt ebenfalls die aus dem derart umstrukturierten Workflow-Graphen abgeleitete Bedienungsanleitung. Im Vergleich zur ursprünglichen Bedienungsanleitung aus Abbildung 1 sind darin nun auch solche Partner erfasst, die zwischen zwei Geboten (`Bid`) nicht eigens auf eine zugehörige Ausschreibung (Nachricht `BidRequest`) warten. Zudem sind die zwei Alternativen zum Beenden des Veräußerungsverfahrens nun explizit wiedergegeben. So kann ein Partner das Verfahren zum einen durch Senden der Nachricht `Abort` abbrechen (vergleiche die durch *!Abort* erreichbaren Zustände). Andererseits ist, durch die eingefügte Unterscheidung möglicher Inhalte der Nachricht `Bid` (*!Bid[assert message(Bid)>1000]*, *!Bid[assert not(message(Bid)>1000)]*), zudem auch das Ende nach einem den Mindestverkaufswert übersteigenden Gebot des Partners repräsentiert.

4 Diskussion und Zusammenfassung

In der vorliegenden Arbeit beschreiben wir einen Ansatz zur Erweiterung der in [4] vorgestellten Umstrukturierungsmethode für verteilte Geschäftsprozesse. Diese erlaubt bestimmte Schleifen und Verzweigungen so zu transformieren, dass deren Bedingungen eliminiert werden können. Auf diese Weise lassen sich die petrinetzbasierten Prozessmodelle bestehender Analysen präzisieren und dadurch genauere Analyseergebnisse erzielen. In [4] werden Bedingungen betrachtet, die nur auf durch Konstanten definierte Variablen zugreifen. Durch die Verwendung eines abstrakteren Begriffs der Schleifeninstanz und die Berücksichtigung von Nachrichteninhalten können wir die Methode nun auch auf Bedingungen anwenden, deren Variablen zusätzlich durch Nachrichten definiert sind.

Im selben Moment stellt sich aber auch die Frage nach der Notwendigkeit einer Präzisierung der Prozessmodelle. So bildet die in Abbildung 1 dargestellte Bedienungsanleitung bereits eine korrekte, wenn auch unvollständige, Beschreibung des von außen sichtbaren Verhaltens der Aktivität `BiddingSequence`. Daher kann der Nutzen einer präziseren Bedienungsanleitung infrage gestellt werden, insbesondere im Hinblick auf die darin veröffentlichte, möglicherweise vertrauliche, Information zum Mindestverkaufswert. Anders gestaltet sich die Situation jedoch, falls ein leicht modifizierter Prozess betrachtet wird. Ein Prozess, der beispielsweise keine Ausschreibungen sendet, dass heißt die Aktivität `Invoke BidRequest` nicht enthält, dafür aber jedes Gebot mit einer entsprechenden Nachricht bestätigt, sollte ebenfalls durch eine Bedienungsanleitung beschrieben werden können. Die Ableitung einer Bedienungsanleitung ist aber in diesem Fall nicht mehr möglich. Stattdessen kann aufgrund der nichtdeterministischen Abbildung der Schleifenbedingung unter Verwendung des herkömmlichen petrinetzbasierten Prozessmodells kein nicht verklemmender Partnerprozess gefunden werden. Eine Präzisierung des Prozessmodells, etwa durch Anwendung der vorgestellten Umstrukturierungsmethode, ist zur Ableitung einer Bedienungsanleitung für diesen Prozess zwingend erforderlich. Gleichzeitig muss diese nun auch die Information zum Mindestverkaufswert der Veräußerung bereitstellen.

In weiterführenden Arbeiten wollen wir den vorgestellten Ansatz zunächst evaluieren. Dazu soll sowohl eine Implementierung als auch eine Sammlung realistischer Beispielprozesse zur Beurteilung der praktischen Relevanz der Umstrukturierungsmethode verwirklicht werden. In diesem Zusammenhang stellt sich auch die Frage nach der Abgrenzung zu vergleichbaren Ansätzen der Prädikatenabstraktion [3]. Darüber hinaus sind wir, wie bereits in [4] angedeutet, an der schrittweisen Ausweitung des Ansatzes auf andere statisch auswertbare Bedingungen interessiert. Zu diesem Zweck erscheint uns insbesondere der hier eingeführte allgemeinere Begriff der Schleifeninstanz ein geeignetes Mittel. Durch eine präzisere Analyse der Datenabhängigkeiten von Schleifen- und Verzweigungsbedingungen ist die Ableitung entsprechender Zusicherungen denkbar. Als eine mögliche Analyse tritt dabei die in der Arbeit [2] beschriebene abstrakte Interpretation der in WS-BPEL-Prozessen genutzten XPath-Ausdrücke zur Abschätzung der Wertebereiche von Variablen hervor.

Literatur

[1] Favre, Cédric: *Algorithmic verification of business process models*, École Polytechnique Fédérale de Lausanne, Master's Thesis, 2008

[2] Görlach, Katharina: *Ein Verfahren zur abstrakten Interpretation von XPath-Ausdrücken in BPEL-Prozessen*, Humboldt-Universität zu Berlin, Diplomarbeit, 2008

[3] Graf, Susanne ; Saidi, Hassen: Construction of Abstract State Graphs with PVS. In: Grumberg, Orna (Hrsg.): *Computer Aided Verification, 9th International Conference, CAV'97, Haifa, Israel, June 22-25, 1997, Proceedings*, Springer-Verlag, 1997 (Lecture Notes in Computer Science 1254), S. 72–83

[4] Heinze, Thomas S. ; Amme, Wolfram ; Moser, Simon: A Restructuring Method for WS-BPEL Business Processes Based on Extended Workflow Graphs. In: Dayal, Umeshwar (Hrsg.) ; Eder, Johann (Hrsg.) ; Koehler, Jana (Hrsg.) ; Reijers, Hajo A. (Hrsg.): *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009, Proceedings*, Springer-Verlag, 2009 (Lecture Notes in Computer Science 5701), S. 211–228

[5] Lohmann, Niels: A Feature-Complete Petri Net Semantics for WS-BPEL 2.0. In: Dumas, Marlon (Hrsg.) ; Heckel, Reiko (Hrsg.): *Web Services and Formal Methods, 4th International Workshop, WS-FM 2007, Brisbane, Australia, September 28-29, 2007*, Springer-Verlag, 2007 (Lecture Notes in Computer Science 4937), S. 77–91

[6] Lohmann, Niels ; Massuthe, Peter ; Stahl, Christian ; Weinberg, Daniela: Analyzing Interacting WS-BPEL Processes Using Flexible Model Generation. In: *Data & Knowledge Engineering* 64 (2008), Nr. 1, S. 38–54

[7] Lohmann, Niels ; Massuthe, Peter ; Wolf, Karsten: Operating Guidelines for Finite-State Services. In: Kleijn, Jetty (Hrsg.) ; Yakovlev, Alex (Hrsg.): *Petri Nets and Other Models of Concurrency - ICATPN 2007, 28th International Conference on Applications and Theory of Petri Nets and Other Models of Concurrency, ICATPN 2007, Siedlce, Poland, June 25-29, 2007, Proceedings*, Springer-Verlag, 2007 (Lecture Notes in Computer Science 4546), S. 321–341

[8] *Web Services Business Process Execution Language Version 2.0.* OASIS Standard, Organization for the Advancement of Structured Information Standards, 2007

[9] *Business Process Model and Notation (BPMN) Version 2.0.* OMG Standard, Object Management Group / Business Process Management Initiative, 2009