Workflow-Operationen als Bereicherung funktionaler Dekomposition in Workflow-Schemata

Michael Schlundt[1], Ralf Schamburger[1], Jens Neeb[1], Stefan Jablonski[1], Markus Böhm[2]

Universität Erlangen-Nürnberg[1]
Lehrstuhl für Datenbanksysteme
Martensstraße 3
91058 Erlangen
{schlundt, schamburger, neeb, jablonski}@informatik.uni-erlangen.de

Technische Universität Dresden[2]
Lehrstuhl für Datenbanken
Dürerstraße 26
01237 Dresden
markus.boehm@inf.tu-dresden.de

1 Einleitung und Motivation

Geschäftsprozesse können durch Workflow-Management-Systeme (WFMS) unterstützt werden, wenn die Arbeitsabläufe in einer für das WFMS interpretierbaren Form vorliegen. Ausführbare Beschreibungen von Arbeitsabläufen bezeichnet man als Workflow-Schemata, diese können mit Hilfe einer Workflow-Sprache realisiert werden. Der Einsatz einer Sprache zur Darstellung eines Sachverhalts wird als Modellieren bezeichnet; mit der Workflow-Sprache wird also ein Modell des Arbeitsablaufs (konzeptionelles Workflow-Schema) erstellt [JBS97].

Generell kann man festhalten, daß es alternative Vorgehensweisen zur Erstellung eines Modells gibt. Meist folgen daraus verschiedene Workflow-Schemata, die funktional gesehen die gleichen Effekte bewirken. Da Workflow-Schemata als Vorlage für auszuführende Arbeitsabläufe dienen, müssen Workflow-Schemata in erster Linie die Bedürfnisse des Anwenders reflektieren. Der Modellierer muß Workflow-Schemata so gestalten, daß zum einen unzulässige Arbeitsabläufe verhindert werden, zum anderen die notwendige Handlungsfreiheit für den Anwender erhalten bleibt. Diese Ziele beeinflussen sich offensichtlich gegenseitig. Der Modellierer muß daher die Möglichkeit haben, unterschiedliche Modellierungskonzepte und Ausdrucksmittel der Workflow-Sprache auszuwählen, um für eine gegebene Anwendungssituation eine passende Realisierung zu finden.

In diesem Beitrag gehen wir auf zwei unterschiedliche Modellierungsansätze bei der Umsetzung von Geschäftsprozeßtätigkeiten in ausführbare Arbeitsschritte eines konzeptionellen Workflow-Schemas ein. Dabei stellt die Zerlegung eines Arbeitsablaufs in Arbeitsschritte ein allgemeingültiges Mittel dar, das der Strukturierung und Reduzierung der Komplexität des gesamten Arbeitsablaufs dient. Die finale Zerlegung in Arbeitsschritte [LoSc75], die unabhängig von der konkreten Repräsentation im konzeptionellen Workflow-Schema ist, bezeichnet man als funktionale Dekomposition. Dabei werden wir in Kapitel 3 auf die Möglichkeiten eingehen, Arbeitsschritte im konzeptionellen Workflow-Schema einerseits als sogenannte Workflow-Operationen oder andererseits als Subworkflows zu realisieren.

Die abstrakt vorgestellten Modellierungsvarianten werden in Kapitel 3 anhand des folgenden einfachen Beispiels diskutiert: In einem Reisebüro werden Flugreisen durch die Arbeitsschritte bzw. Teilaufgaben “Buchen“ und “Stornieren“ von Reservierungen in einem Buchungssystem realisiert. Beide Teilaufgaben treten in unvorhersehbarer Folge, Wiederholung und Häufigkeit auf und konstituieren einen Teil des gesamten Arbeitsablaufs “Flugreise bearbeiten“. Soll für diesen Arbeitsablauf ein unterstützendes WFMS eingeführt werden, ist eine Integration des Buchungssystems als externe Applikation sinnvoll; beide genannten Arbeitsschritte sollen vom WFMS angeboten werden. In Abbildung 1.1 ist das bereits erwähnte Flugreisebeispiel in groben Zügen dargestellt. Aus der Planungsphase der Geschäftsprozeßunterstützung sind die erforderlichen Arbeitsschritte sowie die einzubindenen Applikationen bekannt. Die auftretenden Arbeitsschritte sind funktional dekomponiert abgebildet.

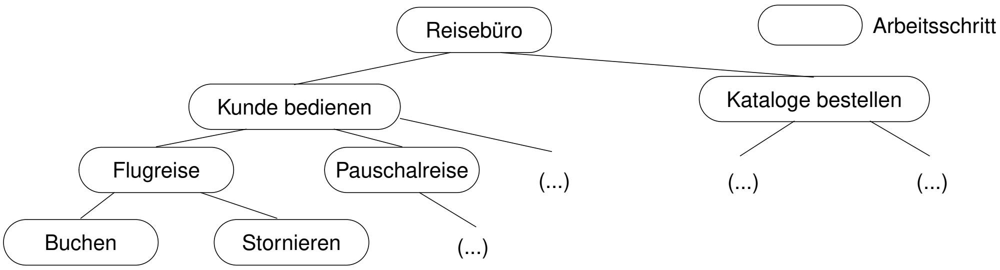

Abb. 1.1: Funktionale Dekomposition eines Geschäftsprozesses im Reisebüro

2 Möglichkeiten der funktionalen Dekomposition

Im Vorgehensmodell zur Entwicklung einer Workflow-Management-Anwendung (WFMA) ist ein hinreichend spezifiziertes Geschäftsprozeßmodell die Basis für die Entwicklung des Workflow-Schemas. In diesem Fachkonzept existiert eine detaillierte Aufgabenstrukturierung, aus der alle Arbeitsabläufe bis einschließlich elementarer Arbeitschritte und ihre Zugehörigkeit zum Gesamtablauf ersichtlich sind. Der darauf folgende Entwicklungsschritt behandelt das Systemkonzept, in dem eine Umsetzung des Geschäftsprozeßmodells in den strengeren Formalismus einer durch das WFMS interpretierbaren Form durchzuführen ist [Ortn97].

Workflow-Sprachen wie [COSA96], [FlMa96] haben für die Umsetzung von funktional dekomponierten Arbeitsabläufen entsprechende Strukturierungsmöglichkeiten. In der diesem Paper zugrunde liegenden Workflow-Sprache, die auf der *MOBILE* Script Language [JaBu96] basiert, setzen wir die Möglichkeit zur Definition von kompositen und elementaren Workflow-Schemata voraus. Ein komposites Workflow-Schema setzt sich aus mehreren anderen (kompositen und/oder elementaren) Workflow-Schemata zusammen. Enthält ein Workflow-Schema keine Verwendung eines weiteren Workflow-Schemas, so handelt es sich um ein elementares Workflow-Schema. Aus der Definition von kompositen und elementaren Workflow-Schemata ergibt sich die Super-Sub-Workflow-Beziehung [JaBu96]: alle Verwendungen von Workflow-Schemata, die ein komposites Workflow-Schema X enthält, werden als Subworkflow von X bezeichnet. Im Gegenzug dazu wird eine Verwendung eines kompositen Workflow-Schemas als Superworkflow von allen Subworkflows, die er enthält, bezeichnet. Verwendungen von Workflow-Schemata werden also im folgenden als Workflows bezeichnet. Abhängig von dem Standpunkt des Betrachters kann ein Workflow gleichzeitig Super- und Subworkflow sein.

Bei der Umsetzung des Geschäftsprozeßmodells in ein Workflow-Schema nur auf Basis funktionaler Dekomposition mit Hilfe von Subworkflows wird dem Modellierer dahingehend ein Korsett aufgezwängt, daß er verschiedene Applikationsaufrufe durch verschiedene Workflows realisieren muß. Dies kann im Besonderen auch bedeuten, daß logisch zusammengehörige Arbeitsschritte (Arbeitseinheiten) des Geschäftsprozeßmodells im Workflow-Schema auf mehrere Subworkflows aufgeteilt werden müssen. Als Folge davon ist der Modellierer gezwungen, jeden beliebigen Freiheitsgrad in der Abarbeitung des Workflows nachzubilden, um eine möglichst hohe Akzeptanz und Benutzerfreundlichkeit des Workflow-Schemas zu erreichen. Daraus wird oft ein sehr fehleranfälliger und komplexer Kontrollfluß entstehen, der vermutlich dem intuitiven Geschäftsprozeßablauf einer logischen Arbeitseinheit nicht mehr entspricht.

Um die gestalterische Freiheit des Modellierers sowie die Anwendungsfreiheit des Endbenutzers zu erhöhen, schlagen wir die Einführung von Workflow-Operationen vor. Durch diese Operationen können in einem Workflowschritt mehrere Applikationen aufgerufen werden, wodurch logische Arbeitseinheiten nicht aufgespalten werden müssen. Weiterhin wird so dem Endbenutzer ein höherer Freiheitsgrad in der Abarbeitung der eigentlichen Arbeit zugestanden, da er Reihenfolge, Querverweise und Nachschlageoperationen usw. frei ausführen und an seinen Arbeitsstil anpassen kann. So kann eine Gängelung des Mitarbeiters durch vorbestimmte Abarbeitungsreihenfolgen des Workflows durch das WFMS verhindert werden.

Diesem Artikel liegt der folgende Operationsbegriff zugrunde:

- In jedem Workflow-Schema können mehrere Workflow-Operationen definiert werden, die entsprechend ihrer Definition bei der Verwendung unterschiedliche Arbeitsschritte (z.B. Applikationsaufrufe, Pausieren der Workflowausführung usw.) ausführen.
- Es besteht keine Reihenfolgebeziehung zwischen den Workflow-Operationen eines Workflow-Schemas.
- Einem dem Workflow zugeordneten Benutzer werden alle Workflow-Operationen dieses Workflows angeboten.
- In jedem Workflow-Schema existiert mindestens eine Workflow-Operation, welche die Ausführung des Workflows beendet

Zur Vereinfachung der in diesem Artikel vorgestellten Modellierungsproblematik wurde im Vergleich zu [JaBu96] eine eingeschränkte Definition von Workflow-Operationen verwendet, die keine Reihenfolgebeziehungen zwischen Workflow-Operationen zuläßt.

Mit Hilfe der funktionalen Dekomposition von Workflows in Subworkflows und den Workflow-Operationen ergeben sich verschiedene Möglichkeiten, die Teilaufgaben des Geschäftsprozesses auf ein entsprechendes Workflow-Schema abzubilden.

3 Modellierungsvarianten

In diesem Kapitel beschreiben wir die zwei unterschiedlichen Modellierungsvarianten, die sich durch Verwendung von Subworkflow- und Workflow-Operations-Definition bzw. ausschließlicher Strukturierung in Subworkflows ergeben. Zunächst wird auf die Möglichkeit eingegangen, ein Workflow-Schema ausschließlich in Subworkflows funktional zu dekomponieren (Abschnitt 3.1). Eine Lösung, die zusätzlich Workflow-Operationen aufnimmt, wird in Abschnitt 3.2 beschrieben.

3.1 Funktionale Dekomposition in Subworkflows

Im folgenden wird das oben eingeführte Reisebürobeispiel als Workflow WFS_FLUGREISE skizziert. Es wird dabei vorausgesetzt, daß die zugrunde liegende Workflow-Sprache lediglich funktionale Dekomposition als Strukturierungsmöglichkeit anbietet. Kontrollflußkonstrukte, die das Modellieren von temporalen und kausalen Abhängigkeiten erlauben, wie Sequenz, Parallelität, bedingte Verzweigung und Schleife, werden ebenfalls vorausgesetzt.

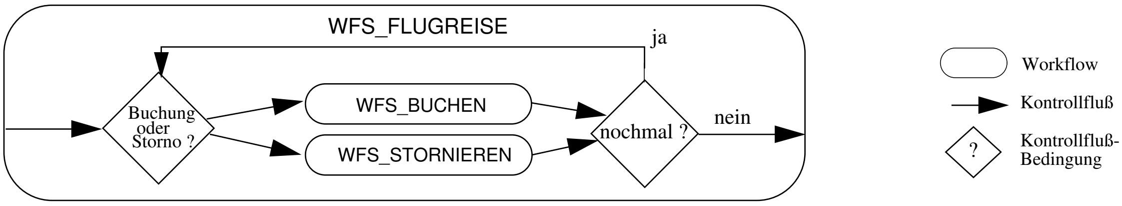

Abb. 3.1: Funktionale Dekomposition in Subworkflows

Die Umsetzung des in Kapitel 1 geschilderten Anwendungsbeispiels gemäß dieser Modellierungsvariante läßt sich wie folgt charakterisieren:

- Durch die Zuordnung einer Applikation zu einem elementaren Workflow sind die Aktionen des Sachbearbeiters aus Sicht des WFMS protokollierbar.
- Eine Applikation wird beim Start eines elementaren Workflows aufgerufen.
- Die Ausführungsreihenfolge der Teilaufgaben wird durch eine explizite Formulierung des Kontrollflusses festgelegt, der die Abhängigkeiten zwischen Subworkflows beschreibt.
- Der Anwender muß vor der Ausführung einer Teilaufgabe angeben, ob eine Buchung oder eine Stornierung durchgeführt werden soll. Nach der Erledigung einer Teilaufgabe erfolgt eine erneute Abfrage, die regelt, ob der Workflow beendet wird, oder eine weitere Buchung bzw. Stornierung durchgeführt werden soll. Damit ist ein explizites Auslassen oder Bestätigen von Teilaufgaben notwendig.

Die Möglichkeit, Aufgaben in elementare Workflows zu unterteilen, wird durch Anwendungsprogramme beschränkt, mit deren Hilfe bestimmte Teilaufgaben realisiert werden sollen. Es ist nicht möglich, im Workflow-Schema eine Aufgabe in weitere Teilaufgaben zu unterteilen, wenn die zu integrierende Applikation diese Aufgabe in einem Schritt erledigt. Allerdings besteht die Möglichkeit ein Workflow-Schema zu erstellen, in dem verschiedene elementare Workflows dieselbe Applikation verwenden, weil mit ihrer Hilfe z.B. sehr viele logische Arbeitseinheiten erfüllt werden, die Applikation aber technisch nicht feiner ansteuerbar ist. Dieses Szenario ist aber aus Sicht der Benutzerfreundlichkeit und Protokollierbarkeit der tatsächlich ausgeführten Tätigkeit nicht wünschenswert.

Werden mehrere Applikationen einem Workflow zugeordnet, so muß entweder der Kontrollfluß zwischen diesen explizit geregelt werden, was aus Anwendersicht zu einer zur Abbildung 3.1 analogen Lösung führt, oder die Koordination der Applikationen wird durch eine weitere Applikation geregelt. Dies führt allerdings im allgemeinen Fall dazu, daß eine Möglichkeit der Protokollierung seitens des WFMSs nicht mehr gegeben ist und stellt damit keine gleichwertige Realisierung dar. Aus diesem Grund ist auch der Aufruf einer “Auswahl-Applikation”, die wiederum die Ausführung der Buchen- oder Stornieren-Anwendung erlaubt, im allgemeinen nicht möglich [Böhm97].

3.2 Operationen als Ergebnis der funktionalen Dekomposition

Wir stellen nun eine Modellierungsvariante vor, in der auftretende Teilaufgaben des Arbeitsprozesses entweder als Workflow-Operation oder als Subworkflow im Workflow-Schema definiert werden können. Im Flugreisebeispiel müssen die Teilaufgaben “Flugreise bearbeiten”, “Buchen” und “Stornieren” in der zur Verfügung stehenden Workflow-Sprache spezifiziert werden.

Gemäß der Definition von Workflow-Operationen in Kapitel 2 ist es möglich, daß für ein Workflow-Schema mehr als nur eine Workflow-Operation spezifiziert wird. Mit Hilfe mehrerer Workflow-Operationen kann somit ein Workflow-Schema WFS_FLUGREISE erstellt werden, das die Workflow-Operationen OP_BUCHEN, OP_STORNIEREN und OP_BEENDEN anbietet (Abbildung 3.2). Ein explizites Beenden des Workflows ist in dem beschriebenen Anwendungsbeispiel notwendig, da die Anzahl der Buchungs- und Stornierungsoperationen im konkreten Anwendungsfall nicht feststeht.

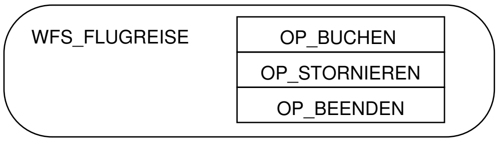

Abb. 3.2: Subworkflow Flugreise mittels Workflow-Operationen

Ein Strukturierung in Subworkflows findet in dieser Realisierungsvariante nur bis zur Ebene des Arbeitsschritts "Flugreise bearbeiten" statt. Somit ist WFS_FLUGREISE ein elementares Workflow-Schema, das drei Workflow-Operationen, nämlich OP_BUCHEN, OP_STORNIEREN und OP_BEENDEN anbietet. Die Ausführungssemantik der drei Workflow-Operationen ist wie folgt definiert: Die Workflow-Operationen OP_BUCHEN und OP_STORNIEREN belassen den Workflow in seinem Zustand und können somit beliebig oft ausgeführt werden, da sie dem Benutzer immer wieder neu zur Verfügung gestellt werden. Die Workflow-Operation OP_BEENDEN terminiert die Ausführung des Workflows.

Die Realisierung des Anwendungsbeispiels aus Kapitel 1 läßt sich wie folgt beschreiben:

- Die Teilaufgaben "Buchen" und "Stornieren" werden jeweils durch einen Workflow-Operationsaufruf getätigt und sind somit für das WFMS protokollierbar.
- Der Applikationsaufruf erfolgt explizit durch das Anstoßen der jeweiligen Workflow-Operation.
- Es existiert keine Ausführungsreihenfolge der Teilaufgaben "Buchen" und "Stornieren". Insbesondere können diese Teilaufgaben mehrfach hintereinander durchgeführt werden, bis die Workflow-Operation OP_BEENDEN aufgerufen wird,.

Dieser Modellierungsansatz ordnet somit nicht jede Teilaufgabe einem elementaren Workflow-Schema zu, sondern realisiert mehrere Teilaufgaben durch unterschiedliche Workflow-Operationen in einem elementaren Workflow-Schema.

4 Entscheidungskriterien zur Bewertung der Modellierungsvarianten

Im folgenden werden Kriterien (abgeleitet aus [Somm96]) diskutiert, anhand derer eine Bewertung der beiden Modellierungsvarianten vorgenommen werden kann. Die reine funktionale Dekomposition in Subworkflows ohne die Möglichkeit unterschiedliche Workflow-Operationen definieren zu können, wird dabei als Subworkflow-Variante und die in Abschnitt 3.2 vorgestellte Modellierungsart als Operations-Variante bezeichnet.

- **Ergonomie der Anwendung aus Benutzersicht**. Das eingeführte Beispiel macht deutlich, daß Anwendungsfälle existieren, für die eine Realisierung nach der Operations-Variante zu einer ergonomischeren und intuitiveren Bedienung führt, da andernfalls vor dem Anstoßen der Aufgabe zunächst die Entscheidung über den weiteren Fortschritt getroffen werden muß (insbesondere betrifft dies die Frage, ob eine Wiederholung der Teilaufgaben "Buchen" und "Stornieren" durchgeführt werden soll).
- **Freiheitsgrade der Benutzung**. Die Operations-Variante läßt dem Anwender die Freiheit, Reihenfolge und Häufigkeit der auszuführenden Teilaufgaben situationsgerecht auszuwählen. Das bedeutet gleichzeitig, daß nach der Definition von Workflow-Operationen in Kapitel 2 reihenfolgeabhängige oder anderweitig bedingte Teilaufgaben nur mittels der Subworkflow-Variante und expliziter Definition der Abhängigkeiten zwischen den Workflows (z.B. Kontrollfluß) zu realisieren sind.
- **Logische Zusammengehörigkeit von Teilaufgaben**. Sind im Fachkonzept Teilaufgaben im gleichen Arbeitskontext zu sehen, die sich noch zusätzlich durch Reihenfolgeunabhängigkeit auszeichnen, so ist dies zu Gunsten der Operations-Variante zu deuten. Als Beispiel können hier vor allem Abläufe dienen, die neben der Haupttätigkeit optionale Nebentätigkeiten wie Informationsbeschaffung durch Nachschlagen beinhalten.

5 Zusammenfassung und Ausblick

In diesem Beitrag zeigen wir, daß zur Abbildung von Teilaufgaben eines Geschäftsprozesses auf ein Workflow-Schema zwei unterschiedliche Modellierungsansätze möglich sind: Die Abbildung von Arbeitschritten auf Subworkflows bzw. die Realisierung dieser Arbeitsschritte als Workflow-Operationen.

Der entscheidende Gewinn bei der Einführung von Workflow-Operationen liegt in der Freiheit des Modellierers, verschiedene Varianten für die Repräsentation von Teilaufgaben zu erhalten. Er kann gemäß den Wünschen des Anwenders oder den Eigenschaften der aufzurufenden Applikationen entscheiden, einen Subworkflow oder eine Operation zur "Einkleidung" dieses Aufrufs zu wählen. Dabei spielen auch nicht-funktionale Kriterien wie Wiederverwendbarkeit, Laufzeitverhalten und Modellierungskomplexität eine Rolle, die in diesem Beitrag nicht diskutiert werden.

Der Unterschied der vorgestellten Modellierungsvarianten wird deutlich, wenn Workflow-Schemata verglichen werden, die einen aus der Anwendungssituation resultierenden hohen Freiheitsgrad erfordern. Steht allein die funktionale Dekomposition in Subworkflows zur Verfügung, kann eine sehr komplexe Kontrollflußlogik oder eine aus Anwendersicht schlechte Ergonomie die Folge sein.

In Workflow-Sprachen, die eine Definition von Workflow-Operationen erlauben, ist eine anwendungsabhängige hybride Modellierung möglich, da durch die Zuordnung von nur einer Workflow-Applikationsoperation zu einem Workflow eine Realisierung, die der Subworkflow-Variante entspricht, erreicht werden kann. Aus diesem Grund stellt eine Workflow-Sprache, welche die Einbindung von Applikation über Workflow-Operationen vorschreibt, keine Einschränkung dar.

Ungeachtet der Tatsache, daß heutige kommerziell erhältliche WFMS das Operations-Konzept noch nicht unterstützen, halten wir diesen Ansatz für eine diskussionswürdige Variante zur Verwendung von Subworkflows. Erste Erfahrung im Prototyp MOBILE sind erfolgversprechend, zeigen aber auch noch Forschungsbedarf [JaBu96]. Beispielsweise ist zu untersuchen, ob es sinnvoll ist, zwischen den Operationen eine Reihenfolge oder Synchronisation einzuführen. Das Operations-Konzept kann zudem noch verfeinert werden, indem die Zuordnung von Operationen zu Aktivitätsträgern flexibler gestaltet wird und Zuordnungsstrategien (Policies) auf Operations-Ebene zugelassen werden. Betrachtungen, die Auswirkungen des Operations-Konzepts auf weitere Aspekte (z.B. Datenaspekt oder Organisationsaspekt) sowie die Anwendung von Techniken aus der Objektorientierung [GaHe95] (z.B. Vererbung von Workflow-Operationen) behandeln, würden den Rahmen dieses Beitrags sprengen und erfolgen daher an dieser Stelle nicht mehr.

6 Literatur

Böhm97 Böhm M.: "Entwurf und Bewertung von Varianten zur Geschäftsprozeßunterstützung mit Workflow-Management-Systemen am Beispiel der Integration externer Anwendungen", in: Workshop "Arbeitsplatzrechner - Integration zur Prozeßverbesserung", Informatik '97, Jahrestagung der GI, Aachen, 1997

COSA96 COSA Workflow Referenz-Handbuch. Version 1.4.2, Software-Ley, 1996.

FlMa96 IBM Flow Mark, Modeling Workflow, Version 2 Release 3, SH 19-8241-01, 1996.

GaHe95 Gamma E., Helm R., Johnson R., Vlissides J.: "Design Patterns: Elements of Reusable Object-Oriented Sofware", Addisoin-Wesley, Reading, 1995

JaBu96 Jablonski S., Bussler C.: "Workflow Management: Modeling Concepts, Architecture and Implementation", International Thomson Computer Press, London, 1996

JBS97 Jablonski S., Böhm M., Schulze W. (Hrsg.): "Workflow-Management: Entwicklung von Anwendungen und Systemen", dpunkt-Verlag, Heidelberg, 1997

LoSc75 Lorenzen P., Schwemmer O.: "Konstruktive Logik, Ethik und Wissenschaftstheorie", Mannheim, 1975

Ortn97 Ortner E.: "Abgrenzung nach außen" in [JBS97], dpunkt-Verlag, Heidelberg, 1997

Somm96 Sommerville I.: "Software Engeneering", 5th Edition, Addision-Wesley, 1996