February 1999

Wechselwirkungen von Abhängigkeiten in transaktionalen Workflows

Kerstin Schwarz
Universität Magdeburg, schwarz@iti.cs.uni-magdeburg.de

Can Türker
Universität Magdeburg, tuerker@iti.cs.uni-magdeburg.de

Wechselwirkungen von Abhängigkeiten in transaktionalen Workflows

Kerstin Schwarz
Universität Magdeburg (schwarz@iti.cs.uni-magdeburg.de)

Can Türker
Universität Magdeburg (tuerker@iti.cs.uni-magdeburg.de)

Inhalt

Electronic Business Engineering / 4. Internationale Tagung Wirtschaftsinformatik 1999.
Hrsg.: August-Wilhelm Scheer; Markus Nüttgens. – Heidelberg: Physica-Verlag, 1999

Abstract

Workflows bestehen aus Tasks mit vielschichtigen Beziehungen untereinander. Beziehungen zwischen wenigen Tasks sind meist offensichtlich, haben aber häufig einen indirekten Einfluß auf andere Tasks. Bereits die Verknüpfung mehrerer Tasks kann zu unüberschaubaren Spezifikationen mit komplexen Wechselwirkungen zwischen den Tasks führen. Deshalb ist es für den Designer oft schwierig, die transitiven Beziehungen zwischen beliebigen Tasks des Workflows zu erkennen und Rückschlüsse bezüglich einer korrekten, der Real-Welt entsprechenden Modellierung zu ziehen. Der Designer benötigt eine Unterstützung des Entwurfprozesses auf konzeptioneller Ebene.

Wir schlagen das Konzept der Transaktionshülle als formales Modell für transaktionale Workflows vor. Die Tasks des Workflows werden auf Transaktionen und die Beziehungen auf Abhängigkeiten zwischen den Transaktionen einer Transaktionshülle abgebildet. Unser Rahmen verfügt über drei Dimensionen von Abhängigkeiten, den Terminierungs-, Ausführungs- und Objektsichtbarkeitsabhängigkeiten. Wir sind in der Lage, beliebige Transaktionen in Beziehung zu setzen, auch wenn keine direkte Abhängigkeit zwischen ihnen spezifiziert wurde. Das Ableiten transitiver Abhängigkeiten erlaubt uns eine Verbesserung des Workflowentwurfs, indem wir überflüssige Tasks und widersprüchliche Abhängigkeiten zwischen Tasks (automatisch) erkennen. Weiterhin können wir Aussagen über die Folgen des Eintretens bestimmter Transaktionsereignisse treffen und mittels einer Regelmenge dem Designer Verbesserungsvorschläge anbieten.

1 Einleitung und Motivation

Workflowsysteme unterstützen die Durchführung von Geschäftsprozessen unter Einsatz von informationstechnischen Mitteln (Jablonski et al. 1996). Geschäftsprozesse werden zunächst modelliert und dann auf Workflowdefinitionen abgebildet. Workflows sind Aktivitäten, die die koordinierte Ausführung einer Reihe von Aufgaben (Tasks) umfaßt (Georgakopoulos et al. 1995, Leymann 1996). Der Begriff des transaktionalen Workflows (Sheth et al. 1993) spiegelt die Relevanz von Transaktionen für Workflows wider (Eder et al. 1996). Transaktionale Workflows beinhalten die koordinierte Ausführung vieler, in Beziehung stehender Tasks, die auf heterogene, autonome und verteilte Systeme zugreifen und transaktionale Eigenschaften der Tasks des Workflows ausnutzen (Worah et al. 1997).

Zur Spezifikation der Korrektheit des Workflows, der Datenkonsistenz und der zuverlässigen Ausführung von Workflows werden in der Regel erweiterte Transaktionsmodelle (Elmagarmid 1992) verwendet. Erweiterte Transaktionsmodelle sind meistens auf ein bestimmtes Anwendungsgebiet, wie z.B. Dokumentenverwaltung, oder einen bestimmten Problembereich, wie z.B. die Telekommunikati-

on, zugeschnitten und fordern Restriktionen für die Eigenschaften ihrer Transaktionen mit dem Ziel, möglichst viele der klassischen ACID-Eigenschaften der Transaktionen zu garantieren. Transaktionale Workflows verlangen nicht, daß Workflows Datenbanktransaktionen entsprechen und alle ACID-Eigenschaften erfüllen. Sie müssen nicht zwingend Transaktionsmerkmale aufweisen, die von TP-Monitoren (z.B. Serialisierbarkeit und Recovery) unterstützt werden. Trotzdem teilen transaktionale Workflows die Ziele erweiterter Transaktionsmodelle in der Hinsicht, daß sie eine abgeschwächte Transaktionssemantik für eine Menge von Aktivitäten erzwingen. Insgesamt verlangen Modelle für transaktionale Workflows (Georgakopoulos et al. 1995, Reuter et al. 1997, Alonso et al.1996) einen allgemeineren Rahmen zur Beschreibung von in Beziehung stehenden Tasks, die zusammen den Workflow bilden.

Transaktionale Workflows bestehen aus einer großen Anzahl von Tasks, deren Zusammenspiel sehr komplex sein kann. Dies führt zu folgenden Problemen: Die Spezifikation wird unübersichtlich und aufgrund der komplexen Struktur sind Zusammenhänge zwischen beliebigen Transaktionen nicht mehr ersichtlich. Zwischen den einzelnen Tasks eines Workflows bestehen vielschichtige Beziehungen, beispielweise Datenfluß- und Kontrollflußabhängigkeiten, die in Kombination schwer darstellbar sind und deren Wechselwirkungen nicht offensichtlich sind. Beispielsweise können Abhängigkeiten so spezifiziert worden sein, daß bestimmte Transaktionen immer abbrechen müssen oder erst gar nicht ausgeführt werden. Um eine korrekte Modellierung eines transaktionalen Workflows und Ableiten von Aussagen über die Folgen des Eintretens bestimmter Transaktionsereignisse treffen zu können, ist eine *Unterstützung des Entwurfsprozesses auf konzeptioneller Ebene* notwendig.

Wir schlagen das Modell der Transaktionshüllen (Schwarz et al. 1998c) vor, das Gesetzmäßigkeiten der Real-Welt als Abhängigkeiten zwischen Transaktionen modelliert. Mit Hilfe dieses einheitlichen und formalen Rahmens können wir Abhängigkeiten zwischen Transaktionen (Tasks) beschreiben, die sowohl in erweiterten Transaktionsmodellen (Moss 1985) als auch in Aktivitätenmodellen (Dayal et al.1991, Rusinkiewicz et al. 1995) oder Workflow-Modellen (Georgakopoulos et al. 1995, Worah et al. 1997) vorkommen. Aufgrund ihrer Flexibilität sind Transaktionshüllen geeignet, die Koordination, Isolation und Atomarität von Tasks eines transaktionalen Workflows zu spezifizieren. Wir bilden die Tasks auf Transaktionen und die Beziehungen zwischen Tasks auf Abhängigkeiten ab. Der Ansatz der Transaktionshüllen bietet drei abgeschlossene Dimensionen von Abhängigkeiten, den Terminierungs-, Ausführungs- und Objektsichtbarkeitsabhängigkeiten.

Terminierungsabhängigkeiten werden definiert, um Einflüsse von Transaktionsereignissen (erfolgreiches Ende oder Abbruch) auf andere Transaktionen zu berücksichtigen. Die Beschreibung des Datenflusses ist mit Hilfe der *Objektsichtbarkeitsabhängigkeiten* möglich. Ausführungsreihenfolgen und -beziehungen werden über *Ausführungsabhängigkeiten* modelliert. Die Abgeschlossenheit der jeweiligen Dimensionen von Abhängigkeiten bedeutet, daß zwischen beliebigen Transaktionen einer Transaktionshülle immer eine Abhängigkeit einer Dimension gelten

muß. Aufgrund der Eigenschaft der Abgeschlossenheit sind wir in der Lage, transitive Beziehungen zwischen Transaktionen innerhalb einer Dimension abzuleiten. Die Grundlage dafür bildet eine Regelmenge. Die transitiven Abhängigkeiten erlauben es, Widersprüche zwischen direkt spezifizierten Abhängigkeiten und den transitiv abgeleiteten Abhängigkeiten aufzudecken. In diesem Papier untersuchen wir Kombinationen von Abhängigkeiten unterschiedlicher Dimensionen und zeigen deren Wechselwirkungen auf. Das Wissen über diese Wechselwirkungen ermöglicht uns das Erkennen widersprüchlicher Abhängigkeiten. Insbesondere können wir Transaktionen ermitteln, die überflüssig in dem Sinne sind, daß sie niemals erfolgreich beendet werden können. Somit sind wir in der Lage, Aussagen über Konsistenz (und zum Teil über die Güte) einer Workflowspezifikation treffen zu können.

Das Papier ist wie folgt aufgebaut. Abschnitt 2 führt das Konzept der Transaktionshüllen einschließlich der unterschiedlichen Dimensionen von Abhängigkeiten ein. In Abschnitt 3 diskutieren wir zum einen die Wechselwirkungen der Abhängigkeiten innerhalb einer Dimension und zum anderen zwischen den Dimensionen. Im Rahmen dieser Untersuchungen werden transitive Abhängigkeiten einer Dimension hergeleitet und Konsequenzen der Kombinationen von Abhängigkeiten unterschiedlicher Dimensionen diskutiert. Abschnitt 4 zeigt, inwieweit ein Workflow durch das Eintreten eines bestimmten Transaktionsereignisses beeinflußt wird und welche Schlüsse gezogen werden können. Insbesondere skizzieren wir einen Ablauf zur korrekten Spezifikation von Abhängigkeiten in transaktionalen Workflows. Den Abschluß bildet eine Zusammenfassung und ein Ausblick auf zukünftige Arbeiten.

2 Das Konzept der Transaktionshülle

Transaktionshüllen (Schwarz et al. 1998c) bestehen aus einer Menge von Transaktionen, die über Vater-Kindbeziehungen baumartig aufgebaut sind. Jede Transaktionshülle hat *genau eine* Wurzeltransaktion. Jede Nicht-Wurzeltransaktion (Kindtransaktion) hat *genau eine* initiierende Transaktion, die Vatertransaktion. Die Initiierung einer Nicht-Wurzeltransaktion muß folglich der Initiierung der Vatertransaktion folgen. Transaktionen einer Transaktionshülle stehen über sogenannte Transaktionsabhängigkeiten in Beziehung.

Transaktionsabhängigkeiten spezifizieren Einschränkungen für das Auftreten von signifikanten Transaktionsereignissen, z.B. Bedingungen über den Beginn (**begin**), den Abbruch (**abort**) oder das erfolgreiche Ende (**commit**) von Transaktionen. Die Ursache für einen Abbruch spielt in diesem Rahmen keine Rolle. Er kann beispielsweise vom Nutzer initiiert werden. Beziehungen zwischen Tasks in Workflows können auf Abhängigkeiten zwischen Transaktionen einer Transaktionshülle abgebildet werden. Die unterschiedlichen Dimensionen der Abhängigkeiten lassen sich aus den Betrachtungen der signifikanten Ereignisse von Transaktionen ableiten. Setzen wir beispielsweise zwei Transaktionen in Beziehung, so verfügt jede Transaktion über ein Startereignis und ein Terminierungsereignis. In

Abbildung 1 sind Beziehungen zwischen den signifikaten Ereignissen zweier Transaktionen t_i und t_j dargestellt. Transaktion t_i terminiert mit einem **abort** und Transaktion t_j mit dem Terminierungsereignis **commit**. Die gestrichelten Linien deuten die Beziehungen zwischen den signifikanten Ereignissen der beteiligten Transaktionen an.

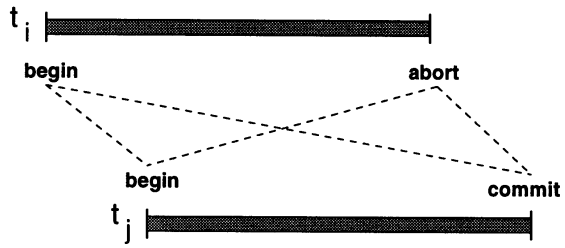

Abbildung 1: Beziehungen zwischen den signifikanten Ereignissen zweier Transaktionen

Wir haben die Wechselwirkungen von Transaktionen bezüglich ihrer signifikanten Ereignisse eingehend untersucht und die drei folgenden Dimensionen von Abhängigkeiten identifiziert:

- Terminierungsabhängigkeiten
- Ausführungsabhängigkeiten
- Objektsichtbarkeitsabhängigkeiten

Terminierungsabhängigkeiten legen die gültigen Terminierungsereignisse der an der Abhängigkeit beteiligten Transaktionen fest. Mit Hilfe der Ausführungsabhängigkeiten sind wir in der Lage, den Kontrollfluß zwischen Transaktionen zu beschreiben. Die Objektsichtbarkeitsabhängigkeiten legen den Datenfluß zwischen Transaktionspaaren fest. In den folgenden Unterabschnitten gehen wir näher auf diese Dimensionen von Abhängigkeiten ein.

2.1 Terminierungsabhängigkeiten

Jede Transaktion kann ihre Ausführung entweder erfolgreich mit **commit** beenden oder mit **abort** abbrechen. Terminierungsabhängigkeiten zwischen zwei unterschiedlichen Transaktionen legen zulässige und unzulässige Kombinationen von Terminierungsereignissen dieser beiden Transaktionen fest (Schwarz et al. 1998c). Für ein Transaktionspaare (t_i, t_j) können wir die folgenden Fälle unterscheiden:

- Beide Transaktionen brechen ab: ($\mathbf{a}_{t_i}$, $\mathbf{a}_{t_j}$).
- Erfolgreiches Ende einer Transaktion und Abbruch der anderen Transaktion: ($\mathbf{a}_{t_i}$, $\mathbf{c}_{t_j}$)/($\mathbf{c}_{t_i}$, $\mathbf{a}_{t_j}$).
- Beide Transaktionen beenden ihre Ausführungen erfolgreich: ($\mathbf{c}_{t_i}$, $\mathbf{c}_{t_j}$).

Diese vier Kombinationen von Terminierungsereignissen können für bestimmte Abhängigkeiten zulässig (∗) oder unzulässig (−) sein. Folglich gibt es sechzehn verschiedene Varianten, die vier möglichen Kombinationen als zulässig bzw. unzulässig auszuzeichnen. Wir treffen hier die Annahme, daß der Abbruch beider Transaktionen immer zulässig ist, weil das **commit** einer Transaktion nicht erzwungen werden kann, z.B. bei einer Integritätsverletzung. Darüber hinaus könnte ein (autorisierter) Benutzer jederzeit einen Abbruch der Transaktion initiieren. Unter dieser Annahme reduziert sich die mögliche Anzahl von Abhängigkeiten auf acht Varianten. Von diesen acht Möglichkeiten haben wir fünf Abhängigkeiten als sinnvoll und geeignet identifiziert. Alle weiteren lassen nur die Kombinationen von Terminierungsereignissen zu, die ausschließlich den Abbruch mindestens einer der beteiligten Transaktionen verlangen und in keinem Fall das erfolgreiche Ende erlauben. Eine Transaktion, die nicht erfolgreich enden darf, muß nach unserer Auffassung nicht gestartet werden. Derartige Transaktionen sind überflüssig, denn abgebrochene Transaktionen hinterlassen keine Effekte in der Datenbank.

Die fünf sinnvollen Abhängigkeiten sind in Tabelle 1 aufgeführt. Zwei Transaktionen werden als *vital-dependent* bezeichnet (**vital_dep**(t_i,t_j)), wenn beide Transaktionen voneinander abbruchabhängig sind, d. h. der Abbruch der einen Transaktion führt zum Abbruch der anderen Transaktion und umgekehrt. Folglich brechen derartig in Beziehung stehende Transaktionen entweder beide ab, oder sind beide erfolgreich.

t_i	t_j	**Vital_dep**(t_i,t_j)	**vital**(t_i,t_j)	**dep**(t_i,t_j)	**exc**(t_i,t_j)	**indep**(t_i,t_j)
$\mathbf{a}_{t_i}$	$\mathbf{a}_{t_j}$	∗	∗	∗	∗	∗
$\mathbf{a}_{t_i}$	$\mathbf{c}_{t_j}$	−	−	∗	∗	∗
$\mathbf{c}_{t_i}$	$\mathbf{a}_{t_j}$	−	∗	−	∗	∗
$\mathbf{c}_{t_i}$	$\mathbf{c}_{t_j}$	∗	∗	∗	−	∗

Tabelle 1: Sinnvolle Terminierungsabhängigkeiten zweier Transaktionen t_i und t_j

Die vitale Abhängigkeit **vital**(t_i,t_j) legt fest, daß die Transaktion t_i lebenswichtig für die Transaktion t_j in dem Sinne ist, daß t_j nicht erfolgreich sein kann, ohne daß t_i ebenfalls erfolgreich ist. Dagegen ist bei der Abhängigkeit **dep**(t_i,t_j) die Transaktion t_i abbruchabhängig von der Transaktion t_j. Eine exklusive Abhängigkeit (**exc**(t_i,t_j)) erlaubt nur einer der beteiligten Transaktionen erfolgreich zu enden, beispielsweise wenn zwei Alternativtransaktionen für einen bestimmten Zweck (z.B. Flugbuchung) gestartet werden. Eine Beziehung zwischen zwei Transaktionen, die nicht eine der vier zuvor beschriebenen Abhängigkeiten erfüllt, wird als unabhängig (**indep**(t_i,t_j)) bezeichnet. In diesem Fall sind alle Kombinationen von Terminierungsereignissen erlaubt.

Die Abhängigkeiten sind formal wie folgt definiert. Das Pfeilsymbol $\Rightarrow$ beschreibt die logische Implikation, zum Beispiel besagt $(\mathbf{a}_{t_i} \Rightarrow \mathbf{a}_{t_j})$, daß der Abbruch von t_i den Abbruch von t_j erfordert:

$$\begin{aligned}
\mathbf{Vital_dep}(t_i,t_j) &: \Leftrightarrow (\mathbf{a}_{t_i} \Rightarrow \mathbf{a}_{t_j}) \wedge (\mathbf{a}_{t_j} \Rightarrow \mathbf{a}_{t_i}) \\
\mathbf{vital}(t_i,t_j) &: \Leftrightarrow (\mathbf{a}_{t_i} \Rightarrow \mathbf{a}_{t_j}) \\
\mathbf{dep}(t_i,t_j) &: \Leftrightarrow (\mathbf{a}_{t_j} \Rightarrow \mathbf{a}_{t_i}) \\
\mathbf{exc}(t_i,t_j) &: \Leftrightarrow (\mathbf{c}_{t_i} \Rightarrow \mathbf{a}_{t_j}) \vee (\mathbf{c}_{t_j} \Rightarrow \mathbf{a}_{t_i}) \\
\mathbf{Indep}(t_i,t_j) &: \Leftrightarrow \text{True}
\end{aligned}$$

Neben den hier beschriebenen zweistelligen Abhängigkeiten verfügt unser Rahmen zusätzlich über drei weitere Abhängigkeiten, die dreistellig sind (Schwarz et al. 1998b). Denn nur mit einer dreistelligen Abhängigkeit kann beispielsweise ausgedrückt werden, daß eine Transaktion t_k als Folge des Abbruchs zweier weiterer Transaktionen abbricht: $((\mathbf{a}_{t_i} \wedge \mathbf{a}_{t_j}) \Rightarrow \mathbf{a}_{t_k})$. Mit Hilfe der binären und ternären Abhängigkeiten können wir beliebige n-stellige Abhängigkeiten (unter Einsatz sogenannter Dummy-Transaktionen) ausdrücken. Darüber hinaus bleibt der Rahmen mit der geringen Menge an zulässigen Abhängigkeiten überschaubar, so daß wir Schlußfolgerungen aus dem Modellierten ziehen können. Die dreistelligen Abhängigkeiten werden wir aus Platzgründen im folgenden nicht näher betrachten.

2.2 Ausführungsabhängigkeiten

Ausführungsabhängigkeiten (Schwarz et al. 1998a) beschreiben Einschränkungen bezüglich der Reihenfolge von Transaktionen. Reihenfolgebeziehungen zwischen Transaktionen betreffen die signifikanten Ereignisse **begin** (b) und **end** (e) als Verallgemeinerung der Terminierungsereignisse **commit** und **abort**. Die Reihenfolge von Ereignissen wird über das Prädikat $\rightarrow$ festgelegt, beispielsweise bedeutet $(\mathbf{e}_{t_i} \rightarrow \mathbf{b}_{t_j})$, daß das Terminierungsereignis der Transaktion t_i vor dem Startereignis der Transaktion t_j auftreten muß.

Untersuchungen der Zusammenhänge zwischen diesen Ereignissen führten zu den folgenden unterschiedlichen Ausführungsabhängigkeiten zwischen zwei Transaktionen t_i und t_j:

- *Parallel* ($\mathbf{par}(t_i,t_j) :\Leftrightarrow \mathbf{lap}(t_i,t_j) \vee \mathbf{inc}(t_i,t_j)$) als Verallgemeinerung der Abhängigkeiten:
  - *Parallel strikt überlappe*nd:
    $\mathbf{lap}(t_i,t_j) : \Leftrightarrow (\mathbf{b}_{t_i} \rightarrow \mathbf{b}_{t_j}) \wedge (\mathbf{b}_{t_j} \rightarrow \mathbf{e}_{t_i}) \wedge (\mathbf{e}_{t_i} \rightarrow \mathbf{e}_{t_j})$,
  - *Parallel einschließend*: $\mathbf{inc}(t_i,t_j) : \Leftrightarrow (\mathbf{b}_{t_i} \rightarrow \mathbf{b}_{t_j}) \wedge (\mathbf{e}_{t_j} \rightarrow \mathbf{e}_{t_i})$.
- *Sequentiell:* $\mathbf{seq}(t_i,t_j) : \Leftrightarrow (\mathbf{e}_{t_i} \rightarrow \mathbf{b}_{t_j})$ mit den folgenden Spezialfällen, die nur dann gelten, wenn ein bestimmtes Terminierungsereignis betrachtet wird:

- *Sequentiell nach dem* **commit**: **seq_c**$(t_i,t_j) :\Leftrightarrow (\mathbf{c}_{t_i} \rightarrow \mathbf{b}_{t_j})$,
- *Sequentiell nach dem* **abort**: **seq_a**$(t_i,t_j) :\Leftrightarrow (\mathbf{a}_{t_i} \rightarrow \mathbf{b}_{t_j})$.

Die Einteilung der Ausführungsabhängigkeiten in parallel und sequentiell ist naheliegend. Wir unterscheiden im Falle von parallel ausgeführten Transaktionen explizit (zeitlich) überlappende und sich einschließende Transaktionen, wie sie in Abbildung 2 dargestellt sind.

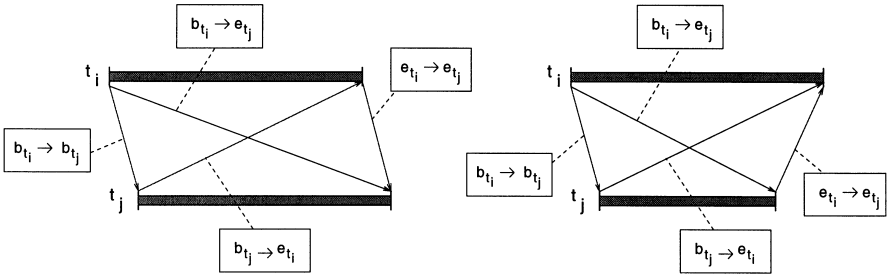

Abbildung 2: Reihenfolge der signifikanten Ereignisse der Abhängigkeiten: lap(t_i,t_j) und inc(t_i,t_j)

Diese Unterscheidung hat den Vorteil, daß parallele Beziehungen differenzierter betrachtet werden können und erlaubt, bessere Schlußfolgerungen zwischen beliebigen Transaktionen einer Transaktionshülle zu ziehen. Insbesondere können Ausführungen von Transaktionen dargestellt werden, die innerhalb des Bereiches einer anderen Transaktion stattfinden (**inc**(t_i,t_j)), wie sie beispielsweise zwischen Vater- und Kindtransaktionen in erweiterten Transaktionsmodellen die Regel sind. Andererseits ist es möglich, eine Transaktion über die zeitlichen Grenzen einer anderen Transaktion hinaus zu modellieren. Ein typischer Fall dafür sind die sogenannten entkoppelten (detached) Transaktionen aus dem Bereich der aktiven Datenbanken (Dayal et al. 1995). Entkoppelte Transaktionen werden innerhalb einer anderen Transaktion initiiert.

Auch im Bereich der Modellierung von Geschäftsprozessen treten parallele Aktivitäten auf, die parallel einschließend oder strikt überlappend sind. Betrachten wir das folgende Beispiel zur Illustration dieser Abhängigkeiten: Ein Automobilhersteller konstruiert eine neues Fahrzeugmodell. Parallel dazu wird bereits vor Beendigung der Konstruktion mit der Bestellung von Rohstoffen und Teilen begonnen und die Produktion gestartet. Die Werbung für das neue Modell beginnt erst nach dem Start der Produktion, aber endet dagegen vor der Einstellung der Produktion dieses Modells (siehe Abbildung 3). In diesem einfachen Beispiel sind also die Konstruktion und Beschaffung bzw. Produktion parallel überlappend, wogegen die Werbung parallel einschließend zur Produktion stattfindet.

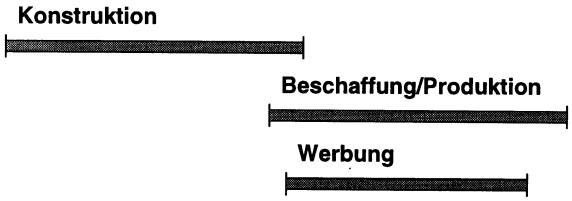

Abbildung 3: Beispiel paralleler Abläufe

Der Einsatz der vorgestellten Ausführungsabhängigkeiten impliziert eine Konjunktion der einzelnen Reihenfolgeterme und der Abhängigkeiten. Die Abhängigkeiten können in unserem Rahmen auch disjunktiv verknüpft werden. Damit entstehen beispielweise Abhängigkeiten wie folgt:

$$(\mathbf{e}_{t_i} \rightarrow \mathbf{b}_{t_k}) \vee (\mathbf{e}_{t_j} \rightarrow \mathbf{b}_{t_k}).$$

Mit Hilfe der disjunktiv verknüpften Ausführungsabhängigkeiten sind wir in der Lage, komplexe Ausführungszusammenhänge und Alternativen darzustellen. Diese Verknüpfungen sind nicht Fokus dieses Artikels und werden deshalb hier nicht näher betrachtet.

2.3 Objektsichtbarkeitsabhängigkeiten

Die dritte Dimension von Abhängigkeiten zwischen Transaktionen betrifft die Sichtbarkeit der Ergebnisse oder Effekte von Transaktionen auf andere Transaktionen (Schwarz et al. 1998c). Objektsichtbarkeit ist von besonderer Wichtigkeit im Zusammenhang mit isolierten und atomaren Transaktionen. Jede Transaktion wird von der Vatertransaktion initiiert und kann bei der Initiierung Ergebnisse von der initiierenden Transaktion empfangen. Anderseits kann eine Transaktion seine Ergebnisse an eine ausgezeichnete Transaktion mit dem **commit** weitergeben, ohne daß diese Ergebnisse für andere Transaktionen sichtbar sind. Werden die Ergebnisse nicht empfangen oder weitergeleitet, so hat eine Transaktion natürlich immer Zugriff auf den aktuellen Datenbankzustand. Zu jeder dieser beiden Kategorien gibt es zwei Objektsichtbarkeitsabhängigkeiten:

- Empfang von Daten bei Initiierung durch Transaktion t_i:
  - *Erbende* (inheriting) Transaktion t_j: **inher**(t_i,t_j),
  - *Nicht-Erbende* (non-inheriting) Transaktion t_j: **non_inher**(t_i,t_j).
- Freigabe von Transaktionsergebnissen durch Transaktion t_j beim **commit**:
  - *Delegierende* (delelgating) Transaktionen geben ihre Ergebnisse mit Hilfe einer sogenannten delegate-Operation an eine andere Transaktion weiter. Diese Ergebnisse sind nur für die empfangende Transaktion t_i verwendbar: **del**(t_i,t_j),

- *Freigebende* (releasing) Transaktionen geben ihre Ergebnisse frei, so daß alle nebenläufigen Transaktionen diese Ergebnisse verwenden können: **rel**(t_i,t_j).

Diese Abhängigkeiten betreffen also die Sicht einer Transaktion auf die verfügbaren Datenbankobjekte. Die Sicht kann sich auf den aktuellen Datenbankzustand beschränken oder erweitert durch die Ergebnisse der initiierenden Transaktionen bzw. erfolgreich beendeter Subtransaktionen sein.

3 Wechselwirkungen zwischen Transaktionsabhängigkeiten

Im vorherigen Abschnitt haben wir drei Dimensionen von Abhängigkeiten vorgestellt, die Terminierungsabhängigkeiten, Ausführungsabhängigkeiten und Objektsichtbarkeitsabhängigkeiten. Jede Dimension besteht aus einer abgeschlossenen Menge von Abhängigkeiten. Bei Terminierungsabhängigkeiten besteht genau eine der fünf vorgestellten Beziehungen zwischen zwei Transaktionen. Wenn zwei Transaktionen nicht parallel ausgeführt werden können, müssen sie sequentiell ausgeführt werden. Es gilt also zwischen jedem Transaktionspaar einer Transaktionshülle jeweils eine der Terminierungs-, Ausführungs- und Objektsichtbarkeitsabhängigkeiten. Diese Abhängigkeiten müssen zwischen Vater- und Kindtransaktion explizit spezifiziert werden. Abhängigkeiten zwischen beliebigen Transaktionen ergeben sich transitiv aus Abhängigkeiten mit benachbarten Transaktionen. Transitive Abhängigkeiten sind Gegenstand des folgenden Unterabschnitts. Hier wird die Ableitung transitiver Abhängigkeiten innerhalb einer Dimension von Abhängigkeiten erläutert und die Konsequenzen diskutiert. Anschließend gehen wir auf Kombinationen von Abhängigkeit zwischen Transaktionen ein, die unterschiedlichen Dimensionen angehören. Die grundlegenden Prinzipien und Probleme verdeutlichen wir anhand einfacher Beispiele.

3.1 Widersprüchliche Abhängigkeiten – Wechselwirkungen innerhalb einer Dimension

Wir bewegen uns zunächst innerhalb einer Dimension von Abhängigkeiten und werden die transitiven Wirkungen des Auftretens von Terminierungsereignissen einer Transaktion auf andere Transaktionen bestimmen. Betrachten wir drei Transaktionen t_i, t_k und t_j. Zwischen den Transaktionen t_i und t_k sei die Abhängigkeit X und zwischen den Transaktionen t_k und t_j die Abhängigkeit Y definiert (siehe Abbildung 4). Uns interessiert insbesondere, wie die Beziehung zwischen den Transaktionen t_i und t_j von den Abhängigkeiten X und Y beeinflußt werden. Dies kann natürlich auf beliebig viele Transaktionen ausgeweitet werden.

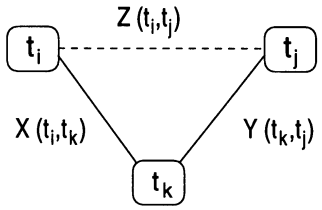

Abbildung 4: Transitive Beziehung zwischen den Transaktionen t_i und t_j

Transitive Abhängigkeiten können innerhalb einer Dimension bestimmt werden. Betrachten wir zunächst zur Verdeutlichung einen Ausschnitt aus einem einfachen Kreditbeispiel bestehend aus den drei Transaktionen t_i, t_k und t_j. Transaktion t_i führt die Risikoabschätzung des Kredits durch, Transaktion t_k überprüft für eine bestimmte Risikogruppe den Kreditrahmen und Transaktion t_j genehmigt den Kredit. Wir abstrahieren in diesem Beispiel von der Ausführungsreihenfolge und legen die folgenden Terminierungsabhängigkeiten fest:

$$\textbf{vital\_dep}(t_i, t_k) \wedge \textbf{vital}(t_k, t_j).$$

Das Szenario ist in Abbildung 5 illustriert. Die Abhängigkeit **vital_dep**(t_i, t_k) verlangt den Abbruch der Risikoabschätzung, wenn die Prüfung des Kreditrahmens abbricht und umgekehrt. Die Abhängigkeit **vital**(t_k, t_j) verlangt, daß kein Kredit genehmigt werden darf (Transaktion t_j), falls der Kreditrahmen nicht erfolgreich geprüft werden konnte. Aus diesem Grund ist der Pfeil in Abbildung 5 von t_k nach t_j gerichtet und der Pfeil zwischen t_i und t_k zu beiden Transaktionen gerichtet.

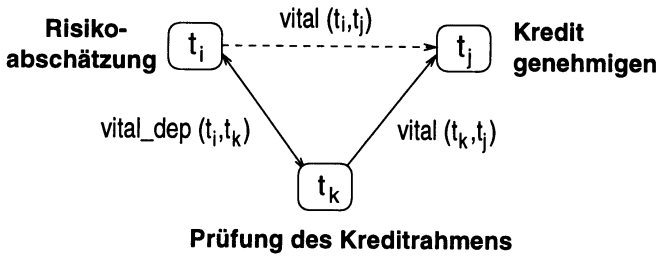

Abbildung 5: Abgeleitete Terminierungsabhängigkeiten

Die transitive Abhängigkeit zwischen den Transaktionen t_i und t_j kann mittels Regeln abgeleitet werden. Diese Regeln haben wir in früheren Arbeiten ermittelt, einen Algorithmus zur Herleitung angegeben und zu einer Regelmenge zusammengefaßt (Schwarz et al. 1998c, Schwarz et al. 1998a). Die folgende Regel kann auf unsere Beispieltransaktionen t_i, t_k und t_j angewendet werden:

$$\textbf{vital\_dep}(t_i, t_k) \wedge \textbf{vital}(t_k, t_j) \Rightarrow \textbf{vital}(t_i, t_j).$$

Die Beziehung zwischen t_i und t_j läßt sich nun einfach ableiten. Bricht Transaktion t_i ab, so bricht t_k und damit t_j ab (Pfeilfolge). In der umgekehrten Richtung gilt dies nicht. Folglich ist die transitive Abhängigkeit **vital**(t_i, t_j), d.h., wenn die Risikoabschätzung nicht erfolgreich durchgeführt werden konnte, wird auch kein Kredit genehmigt. Andererseits kann die Kreditvergabe auch scheitern, wenn die Risikoabschätzung erfolgreich war.

Neben transitiven Terminierungsabhängigkeiten können auch transitive Ausführungs- und Objektsichtbarkeitsabhängigkeiten abgeleitet werden. Ausführungsabhängigkeiten lassen sich dahingehend ermitteln, daß paarweise Abhängigkeiten eine bestimmte Reihenfolge der signifikanten Ereignisse der Transaktionen erfordern. Diese gilt sowohl für die Abhängigkeit X als auch für Y. Aus diesen Reihenfolgeanforderungen lassen sich Anforderungen für die transitive Beziehung Z ableiten.

Betrachten wir das folgende kleine Beispiel der Erstellung eines Angebotes. Transaktion t_i legt die Rahmendaten für das Angebot fest und Transaktion t_k führt die wirtschaftliche Prüfung des Angebotes durch. Die Transaktionen t_i und t_k stehen in einer parallel überlappenden Beziehung **lap**(t_i,t_k). Zusätzlich steht die wirtschaftliche Prüfung mit der rechtlichen Prüfung in Beziehung. Die rechtliche Prüfung wird von Transaktion t_j modelliert und schließt die Transaktion t_k ein (**inc**(t_j,t_k)). In Abbildung 6 sind diese Abhängigkeiten graphisch dargestellt.

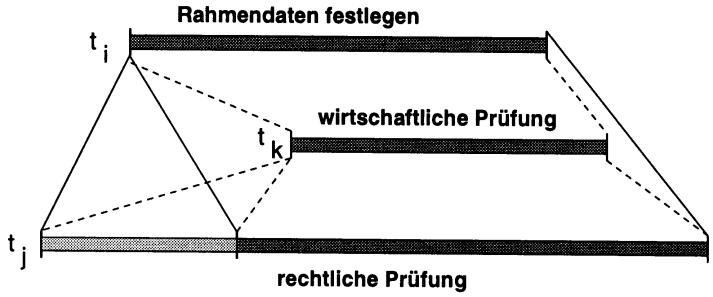

Abbildung 6: Transitive Ausführungsabhängigkeit

Mit Hilfe eines graphentheoretischen Ansatzes lassen sich die Beziehungen zwischen den einzelnen Ereignissen der Transaktionen t_i und t_j ableiten und daraus die transitive Beziehung berechnen (Schwarz et al. 1998a). Betrachten wir die transitive Beziehung zwischen den Transaktionen t_i und t_j im obigen Beispiel. Die Abhängigkeit **lap**(t_i,t_k) verlangt, daß die Transaktion t_i vor der Transaktion t_k terminiert ($\mathbf{e}_{t_i} \rightarrow \mathbf{e}_{t_k}$), und die Abhängigkeit **inc**(t_j,t_k) fordert, daß die Transaktion t_k vor der Transaktion t_j terminiert ($\mathbf{e}_{t_k} \rightarrow \mathbf{e}_{t_j}$). Daraus läßt sich ableiten, daß die Transaktion t_i vor der Transaktion t_j terminiert ($\mathbf{e}_{t_i} \rightarrow \mathbf{e}_{t_j}$). Die Rahmendaten sind also vor dem Ende der rechtlichen Prüfung festgelegt. Für den Beginn der Transaktion t_j (rechtliche Prüfung) bestehen zwei Möglichkeiten. Entweder startet die rechtli-

che Prüfung (t_j) vor der Festlegung der Rahmendaten (t_i) oder danach. Folglich ist die transitive Abhängigkeit zwischen den Transaktionen t_i und t_j entweder parallel überlappend **lap**(t_i,t_j) oder einschließend **inc**(t_j,t_i). Entsprechend der Semantik der Transaktionen in unserem Beispiel wird die rechtliche Prüfung nach dem Beginn der Festlegung der Rahmendaten gestartet werden. Es wird also die Abhängigkeit **lap**(t_i,t_j) geeignet sein.

Wir sehen an diesem einfachen Beispiel, daß dem Designer Hinweise für korrekte Abhängigkeiten gegeben werden können. Falls der Designer der Ansicht ist, daß die vorgeschlagenen Abhängigkeiten nicht korrekt sind, bedeutet dies, daß an anderer Stelle eine andere Abhängigkeit bereits falsch spezifiziert wurde.

Die Objektsichtbarkeitsabhängigkeiten verfügen über zwei grundlegende Transitivitätsregeln. Werden Ergebnisse einer Transaktion t_j an eine andere Transaktion t_i delegiert und gibt letztere Transaktion die Ergebnisse auf die gleiche Weise weiter, so werden die Ergebnisse transitiv delegiert. Gibt eine Transaktion t_i mit der Initiierung von t_k Daten an t_k weiter und verfährt t_k ebenfalls so in bezug auf t_j, so erbt Transaktion t_j transitiv von t_i (auch wenn nicht die gleichen Daten weitergegeben werden, aber Ergebnisse von t_k werden von den Daten von t_i beeinflußt):

$$\mathbf{del}(t_i,t_k) \wedge \mathbf{del}(t_k,t_j) \Rightarrow \mathbf{del}(t_i,t_j),$$

$$\mathbf{inher}(t_i,t_k) \wedge \mathbf{inher}(t_k,t_j) \Rightarrow \mathbf{inher}(t_i,t_j).$$

Alle anderen Kombinationen von Objektsichtbarkeitsabhängigkeiten führen zu nicht vererbenden transitiven Beziehungen und in der anderen Richtung zu freigebenden Transaktionen.

Mit Hilfe unserer Regelmenge können wir die transitive Hülle der Terminierungsabhängigkeiten, Ausführungsabhängigkeiten und Objektsichtbarkeitsabhängigkeiten ableiten. Eine transitiv ermittelte Abhängigkeit muß mit der direkt spezifizierten Abhängigkeit *konsistent* sein. Dies bedeutet, daß die direkte und die transitive Abhängigkeit entweder identisch sind, oder die direkte Abhängigkeit in der Menge der transitiv gültigen Abhängigkeiten enthalten ist. Letzteres kann auftreten, wenn keine eindeutige transitive Abhängigkeit ableitbar ist, sondern eine Menge von gültigen Abhängigkeiten existiert (siehe Abbildung 6). Ist eine direkt spezifizierten Abhängigkeit nicht einer der transitiv ermittelten Abhängigkeiten identisch, so ist die Spezifikation inkonsistent. Wir können folglich Inkonsistenzen in einer Spezifikation erkennen.

Zur Illustration betrachten wir das folgende einfache Beispiel der Buchung einer Konferenzreise: Mit der Transaktion t_i wird eine Person bei der Konferenz angemeldet. Dieser Vorgang besteht aus mehreren Schritten, dem Absenden des Anmeldeformulars, separaten Anmeldungen zu Tutorien und der Anweisung der Konferenzgebühren. Die Transaktion t_k reserviert den Flug und Transaktion t_j bucht ein Hotelzimmer. Transaktion t_i und t_k stehen parallel einschließend in Beziehung wie auch die Transaktionen t_k und t_j. Denn ohne die konkreten Flugdaten kann beispielsweise keine Entscheidung über die Teilnahme an Tutorien getroffen werden. Dieses Szenario ist in der Abbildung 7 über gestrichelte Linien angedeutet. Darüber hinaus wurde bereits zwischen t_i und t_j die direkte Abhängigkeit **seq**(t_i,t_j) spezifiziert (dargestellt durch einen gestrichelten Pfeil). Damit wird ange-

deutet, daß nach der Anmeldung bei der Konferenz ein Hotelzimmer gebucht werden soll. Die Konsistenz dieser Beispielspezifikation läßt sich durch Anwendung der folgenden Regel überprüfen:

$$\mathbf{inc}(t_i,t_k) \wedge \mathbf{inc}(t_k,t_j) \Rightarrow \mathbf{inc}(t_i,t_j).$$

Die ermittelte transitive Abhängigkeit ist also parallel einschließend, die der direkten sequentiellen Abhängigkeit widerspricht. Die Spezifikation ist folglich inkonsistent.

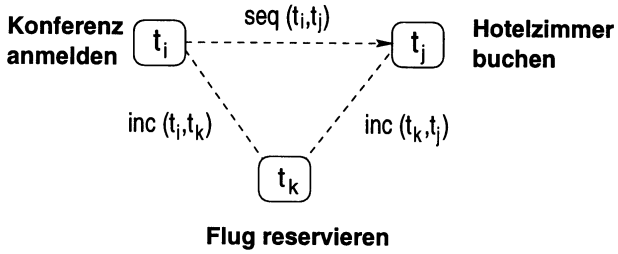

Abbildung 7: Inkonsistente Spezifikation von Ausführungsabhängigkeit

Weitere Inkonsistenzen lassen sich ableiten, wenn Abhängigkeiten unterschiedlicher Dimensionen kombiniert werden, unabhängig davon, ob sie direkt spezifiziert oder transitiv ermittelt wurden. Treten Inkonsistenzen auf, so muß mindestens eine der beteiligten Abhängigkeiten geändert werden, um die Konsistenz herzustellen. Hilfestellungen können mittels der Regeln angeboten werden.

3.2 Überflüssige Spezifikationsteile – Wechselwirkungen zwischen den Dimensionen

Im letzten Unterabschnitt haben wir die Wechselwirkungen von Abhängigkeiten innerhalb einer Dimension betrachtet. Da zwischen Transaktionen jeweils eine Abhängigkeit jeder Dimension entweder direkt oder transitiv gültig ist, gehen wir in diesem Unterabschnitt auf die Kombination der Abhängigkeiten unterschiedlicher Dimensionen ein.

Terminierungsabhängigkeiten legen fest, welche Terminierungsereignisse für in Beziehung stehende Transaktionen gültig sind und welche nicht. Ausführungsabhängigkeiten verlangen dagegen eine Reihenfolge zwischen den auftretenden Ereignissen. Parallel überlappend und einschließend ausgeführte Transaktionen geben insbesondere an, welche der Transaktionen vor der anderen terminiert, wogegen die sequentielle Beziehung festlegt, daß eine Transaktion erst nach der Terminierung einer anderen Transaktion gestartet werden darf. Betrachten wir nun

die Kombinationen der Terminierungsabhängigkeiten mit den Abhängigkeiten sequentiell nach dem Abbruch bzw. nach dem **commit** einer Transaktion.

Die sequentielle Ausführung der Transaktion t_j nach dem Abbruch der Transaktion t_i (**seq_a**(t_i,t_j)) legt fest, daß weder die Ereigniskombination ($\mathbf{c}_{t_i}$, $\mathbf{a}_{t_j}$) noch die Ereigniskombination ($\mathbf{c}_{t_i}$, $\mathbf{c}_{t_j}$) auftreten darf. Dies betrifft die unteren beiden Zeilen der Tabelle 1. Transaktion t_j wird nur nach dem Abbruch von t_i gestartet. Betrachten wir diese Tabelle unter diesem Gesichtspunkt, so bleibt für die Abhängigkeiten **vital_dep**(t_i,t_j) und **vital**(t_i,t_j) nur der Abbruch beider Transaktionen als gültige Ergebniskombination. Folglich sind diese Kombinationen widersprüchlich. Ferner ist die Transaktion t_j überflüssig, da sie erst nach dem Abbruch von t_i gestartet wird und immer abbrechen muß. Diese Transaktion kann aus der Spezifikation entfernt werden oder eine der Abhängigkeiten muß geändert werden.

Besteht dagegen zwischen der Transaktion t_i und der Transaktion t_j die Abhängigkeit **seq_c**(t_i,t_j), so ist die Ereigniskombination ($\mathbf{a}_{t_i}$, $\mathbf{c}_{t_j}$) ungültig. Dabei handelt es sich um die drittletzte Zeile in Tabelle 1. Die kritischen Fälle haben wir in Tabelle 2 dargestellt. Die Kombination **seq_c**(t_i,t_j) und **exc**(t_i,t_j) führt dazu, daß nur der Abbruch von t_j gültig ist. Transaktion t_j ist folglich überflüssig. Ist dagegen die Terminierungsabhängigkeit entweder **vital_dep**(t_i, t_j) oder **dep**(t_i, t_j), so darf Transaktion t_j nach dem erfolgreich Ende von t_i ausschließlich ein **commit** ausführen. Zusätzlich gilt aber aufgrund der sequentiellen Abhängigkeit, daß auch der Beginn der Transaktion t_j dem **commit** von t_i folgen muß. Es ist gemäß unserer Annahmen nicht möglich das **commit** einer Transaktion (hier t_j) zu erzwingen. Somit sind auch diese Kombinationen von Abhängigkeiten widersprüchlich. Das Konzept der Kompensation würde es uns ermöglichen, derartige Kombinationen von Abhängigkeiten zuzulassen. Bei der Kompensation handelt es sich um das semantische Zurücksetzen von Ergebnissen einer Transaktion durch Ausführung einer sogenannten Kompensationstransaktion (Korth et al. 1990).

T_i	t_j	**vital_dep**(t_i,t_j) **seq_c**(t_i,t_j)	**dep**(t_i,t_j) **seq_c**(t_i,t_j)	**exc**(t_i,t_j) **seq_c**(t_i,t_j)
$\mathbf{a}_{t_i}$	$\mathbf{a}_{t_j}$	*	*	*
$\mathbf{a}_{t_i}$	$\mathbf{c}_{t_j}$	–	*	*
$\mathbf{c}_{t_i}$	$\mathbf{a}_{t_j}$	–	–	$(\mathbf{c}_{t_i} \rightarrow \mathbf{a}_{t_j})$
$\mathbf{c}_{t_i}$	$\mathbf{c}_{t_j}$	$(\mathbf{c}_{t_i} \rightarrow \mathbf{c}_{t_j})$	$(\mathbf{c}_{t_i} \rightarrow \mathbf{c}_{t_j})$	–

Tabelle 2: Kombination der Terminierungsabhängigkeiten mit der sequentiellen Abhängigkeit nach dem commit

Betrachten wir nun die Objektsichtbarkeitsabhängigkeiten. Diese Abhängigkeiten beziehen sich auf die Sicht einer Transaktion. In bezug auf delegierende Transaktionen läßt sich folgende Beobachtung feststellen. Werden Ergebnisse einer Transaktion t_j an eine andere Transaktion t_i delegiert, so verlangt diese Operation

eine Terminierung der Transaktion t_j vor der Terminierung der Transaktion t_i. Anderenfalls können keine Ergebnisse weitergereicht werden. Es gilt also wiederum eine Reihenfolgebeziehung zwischen den beteiligten Transaktionen. Diese Reihenfolgebeziehung entspricht einer parallelen Abhängigkeit. Folglich kann die delegierende Eigenschaft nicht mit der sequentiellen Abhängigkeit kombiniert werden (widersprüchliche Kombination). Bezüglich der Terminierungsabhängigkeiten gelten die gleichen Beobachtungen wie bei der Kombination mit der parallelen Abhängigkeit.

Die Widersprüche zwischen Abhängigkeiten unterschiedlicher Dimensionen können direkt oder durch die Ermittlung transitiver Abhängigkeiten aufgedeckt werden. Betrachten wir zur Verdeutlichung das folgende kleine Beispiel. Zwischen den Transaktionen t_i und t_k besteht die Abhängigkeit **seq_c**(t_i,t_k) und zwischen t_k und t_j ebenfalls (siehe Abbildung 8). Die transitive Ausführungsabhängigkeit zwischen t_i und t_j ist folglich **seq_c**(t_i,t_j), die mit der Abhängigkeit **exc**(t_i,t_j) kombiniert wird. Letztere Abhängigkeit ist über einen beidseitig gerichteten Pfeil, der mit einer schrägen Linie versehen ist, dargestellt. Die exklusive Abhängigkeit verlangt, daß eine der beiden Transaktionen abbricht. Wenn Transaktion t_i mit **commit** terminiert, darf aufgrund der sequentiellen Abhängigkeit die Transaktion t_j starten, muß aber aufgrund der exklusiven Abhängigkeit abbrechen. Bricht dagegen Transaktion t_i ab, so darf die Transaktion t_j nicht starten. Transaktion t_j wird also in keinem Fall zu einem **commit** kommen. Transaktion t_j ist folglich überflüssig, es sei denn, der Benutzer verändert eine der beteiligten Abhängigkeiten.

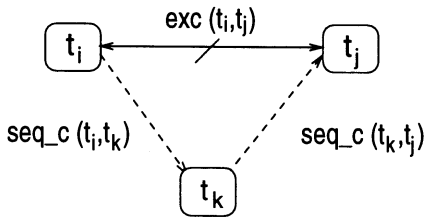

Abbildung 8: Beispiel für eine überflüssige Transaktion t_j

4 Transaktionsereignisse und deren Konsequenzen

Im letzten Abschnitt haben wir gezeigt wie die unterschiedlichen Abhängigkeiten in Beziehung stehen und wie inkonsistente Spezifikationen, also Widersprüche in der Spezifikation und überflüssige Spezifikationsteile, erkannt werden können. Darüber hinaus sind wir in der Lage, die Folgen des Eintretens eines Transaktionsereignisses auf die gesamte Transaktionshülle anzugeben. Betrachten wir beispielsweise den Abbruch einer Transaktion t_i. Alle Transaktionen, die zu t_i in einer vitalen oder vital-dependent Abhängigkeit stehen, sei es nun direkt oder transitiv, müssen ebenfalls abbrechen. Führt die Transaktion t_i dagegen ein **commit** aus, so

werden alle Transaktionen, mit denen sie in einer exklusiven Beziehung steht, abbrechen.

Des weiteren können Folgen bezüglich der Ausführungsabhängigkeiten abgeleitet werden. Besteht beispielsweise zwischen Transaktion t_i und t_j die sequentielle Abhängigkeit **seq_c**(t_i,t_j) und bricht t_i ab, so wird t_j nicht gestartet. Wir können also Schlußfolgerungen auf Grundlage der direkten und transitiven Abhängigkeiten ziehen und diese dem Benutzer offen legen. Unser Ansatz basiert auf einem formalen Modell. Formale Methoden bieten die Grundlage zur Verifikation einer Spezifikation. Unser Ansatz bietet folglich die Möglichkeit, ohne die Ausführung des Workflows simulieren zu müssen, Aussagen über bestimmte Fehler und Ereignisse treffen zu können. Das Testen des modellierten Workflows kann zwar die Anwesenheit von Fehlern zeigen, wir können aber zusätzlich auf Grundlage unserer Regelmenge Verbesserungsvorschläge anbieten.

Sind darüber hinaus Änderungen des Workflows notwendig, können mit unserem Verfahren die Folgen auf den gesamten Workflow (der Transaktionshülle) genau berechnet werden. Des weiteren kann eine kleine Änderung, z.B. einer Terminierungsabhängigkeit von **vital**(t_i,t_j) nach **vital_dep**(t_i,t_j), weitreichende Folgen haben. In unserem Beispiel haben wir mit der Änderung festgelegt, daß der Abbruch der Transaktion t_j den Abbruch von t_i zur Folge hat. Letzteres kann wiederum Auswirkungen auf andere Transaktionen haben. Mit unserem Verfahren können derartige Folgen abgeschätzt werden. Wird beispielsweise eine Verschärfung der Abhängigkeit **seq**(t_i,t_j) zur Abhängigkeit **seq_c**(t_i,t_j) vorgenommen, darf Transaktion t_j nur nach dem **commit** von t_i starten. Im Falle des Abbruchs von t_i wird folglich t_j nicht gestartet und damit gegebenenfalls weitere Teile der Spezifikation nicht ausgeführt, die von Transaktion t_j abhängen. Auch die Folgen derartiger Änderungen können angegeben werden.

Wir gehen davon aus, daß ein Workflow in der Regel aus vielen Tasks besteht. Die hohe Komplexität eines Workflows führt schnell zur Unübersichtlichkeit. Um die Semantik des Workflows vollständig zu erfassen und zu verstehen, leiten wir die transitiven Abhängigkeiten und Konsequenzen bestimmter Ereignisse während des Spezifikationsprozesses ab. Folglich können Übereinstimmungen und Diskrepanzen mit dem realen Geschäftsprozeß schneller und einfacher aufgedeckt werden.

Die Modellierung eines Workflows mittels Transaktionshüllen ist insgesamt wie folgt aufgebaut. Es handelt sich hierbei um einen iterativen Vorgang, der in Abbildung 9 graphisch dargestellt ist. Die grau unterlegten Felder stellen programminterne Berechnungen und Visualisierungen dar und alle weiß unterlegten Felder entsprechen Benutzereingaben. Im folgenden sind die einzelnen Schritte des Designprozesses erläutert:

- Im einzelnen werden zunächst die einzelnen Tasks als Transaktionen durch den Benutzer modelliert.
- Anschließend gibt der Benutzer Beziehungen, also Abhängigkeiten, zwischen den Transaktionen an. Wenn nicht weitere Transaktionen definiert werden

sollen, kann zur Prüfung der Konsistenz der bisherigen Spezifikation übergegangen werden.

- Die Konsistenzprüfung spaltet sich in drei Teile auf. Zunächst werden die transitiven Abhängigkeiten intern vom System ermittelt. Dann wird die Spezifikation nach Widersprüchen zwischen den direkt spezifizierten Abhängigkeiten und den transitiv ermittelten Abhängigkeiten der gleichen Dimension untersucht. Grundlage für die Ermittlung bildet eine Regelmenge (wie in Abschnitt 2 dargestellt). Sind Widersprüche vorhanden, werden sie angezeigt und mögliche Alternativspezifikationen angegeben. Der Workflowdesigner muß nun die Widersprüche korrigieren und die Untersuchung beginnt von neuen.
- Werden dagegen keine Widersprüche innerhalb einer Dimension erkannt, kann entweder mit der Spezifikation fortgefahren werden (zurück zu den ersten Vorgängen) oder die Konsistenz der Spezifikation zwischen den Dimensionen weiter untersucht werden. Letzteres bezieht sich auf die Suche nach Widersprüchen zwischen den direkten oder transitiven Abhängigkeiten unterschiedlicher Dimensionen. Auch in diesem Fall können die Widersprüche und überflüssige Spezifikationsteile erkannt werden. Sind Widersprüche vorhanden, werden diese wiederum angezeigt und der Designer muß ebenfalls eine Korrektur durchführen.
- Wenn alle Widersprüche aus der Spezifikation entfernt sind, gibt es zwei Möglichkeiten fortzufahren. Entweder erweitert der Designer die Spezifikation, oder die Konsequenzen bestimmter Ereignisse werden ermittelt. Dies wird nutzergesteuert durch Markierung von Transaktionen und Belegung mit gewünschten Ereignissen oder vollautomatisch durchgeführt. Es werden dann Spezifikationsteile angezeigt, die nicht ausgeführt werden oder abbrechen, wenn bestimmte Ereignisse eintreten.

Der Einsatz von Transaktionshüllen zur Modellierung von transaktionalen Workflows erlaubt einen korrekten Entwurf mit Hilfestellungen für den Designer. Ein korrekter Entwurf eines Workflows bildet die Grundlage für ein effektives Workflow-Management. Eine widersprüchliche Spezifikation kann dazu führen, daß die Ausführung des Workflows in eine Deadlock-Situation gerät, zum Beispiel bei zyklisch definierten Ausführungsabhängigkeiten:

$$\mathbf{seq}(t_1,t_2) \wedge \mathbf{seq}(t_2,t_3) \wedge \mathbf{seq}(t_3,t_1).$$

Darüber hinaus werden bestimmte Ergebnisse nie erzeugt, wenn sie von Teilen eines Workflows bearbeitet werden, die nie ausgeführt werden oder immer abbrechen müssen (nicht erfolgreich enden dürfen). Letztere Transaktionen schränken zusätzlich die Parallelität und damit die Effizienz des Anwendungssystems ein. Mit unserem Ansatz können derartige Probleme bereits während des Entwurfsprozesses und damit vor der Inbetriebnahme des Workflows beseitigt werden.

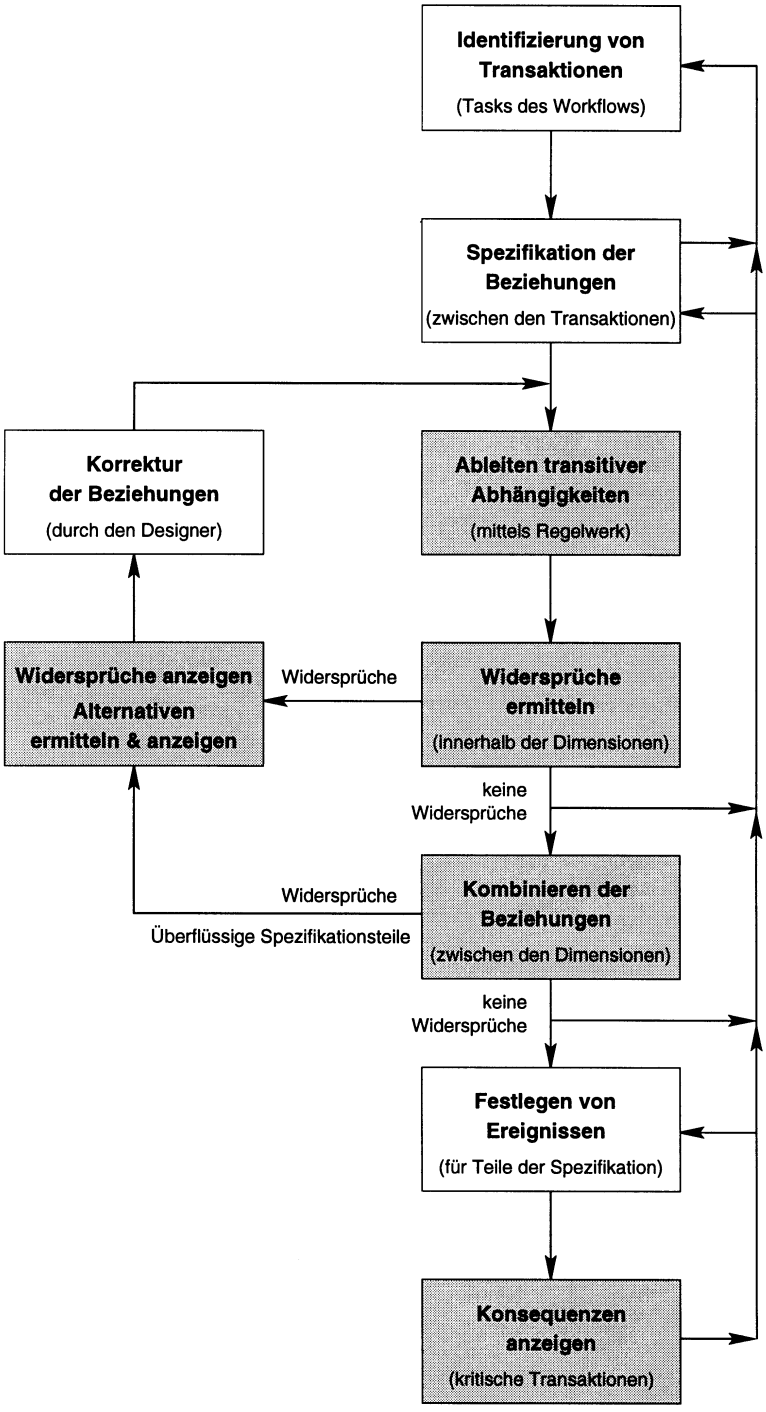

Abbildung 9: Ablauf des Workflowdesigns

5 Zusammenfassung und Ausblick auf zukünftige Arbeiten

Mit zunehmender Komplexität von Anwendungen, die als Workflows modelliert und ausgeführt werden, wird die Notwendigkeit von Transaktionsfunktionalität in Workflow-Management-Systemen deutlich. In dieser Arbeit haben wir das Konzept der Transaktionshüllen als formales Modell für transaktionale Workflows vorgestellt. Zwischen Tasks eines Workflows bestehen vielschichtige Beziehungen, die durch Abhängigkeiten zwischen Transaktionen einer Transaktionshülle ausgedrückt werden können. Unser Spezifikationsrahmen verfügt über drei Dimensionen von Abhängigkeiten, den Terminierungs-, Ausführungs- und Objektsichtbarkeitsabhängigkeiten. Die Abhängigkeiten einer Dimension sind so gewählt, daß zwischen Transaktionspaaren immer eine der Abhängigkeiten gelten muß. Das Ableiten transitiver Abhängigkeiten erlaubt uns eine Verbesserung des Entwurfs, indem wir überflüssige und widersprüchliche Spezifikationsteile erkennen können. Mittels einer Regelmenge sind wir außerdem in der Lage, dem Designer Verbesserungsvorschläge anzubieten. Darüber hinaus können wir Aussagen über die Konsequenzen des Eintretens von Ereignissen auf die gesamte Transaktionshülle treffen. Ein korrekter Entwurf eines Workflows bildet die Grundlage für ein effektives Workflow-Management neben einer schnelleren Umsetzung und kürzeren Zeitspanne bis zum praktischen Einsatz.

In zukünftigen Arbeiten möchten wir uns mit dem Echtzeitaspekt von Transaktionen befassen und in unseren Rahmen integrieren. Damit wären wir in der Lage, einen Ausführungszeitpunkt festzulegen oder einen spätest erlaubten Endzeitpunkt (Deadline) für Transaktionen zu berücksichtigen. Transaktionen, die mit einer Deadline versehen sind, werden abgebrochen, wenn die Zeitschranke nicht eingehalten werden kann. Derzeit beschäftigen wir uns mit der Entwicklung eines Tools und einer geeigneten graphischen Repräsentation der Abhängigkeiten. Die Ableitungsregeln für die einzelnen Dimensionen von Abhängigkeiten sind bereits implementiert worden.

Danksagung

Wir bedanken uns bei Mag. K. J. Fellner und Dr. S. Conrad für hilfreiche Kommentare und Diskussionen.

Literaturverzeichnis

Alonso, G./Agrawal, D./Abbadi, A.E./Kamath, M./Günthör, R./Mohan, C. (1996): Advanced Transaction Models in Workflow Context. In: *Proc. of the 12th Int. Conf. on Data Engeneering*, New Orleans, Louisana, USA, 1996, S. 574-581.

Dayal, U./Hsu, M. Ladin, R. (1991): A Transaction Model for Long-Running Activities. In: Lohmann, G.M. et al. (Hrsg.): *Proc. of the 17th Int. Conf. on Very Large Data Bases (VLDB'91)*, Morgan Kaufmann Publishers, San Mateo, CA, 1991, S. 113-122. Dayal, U./Hanson, E./Widom, J. (1995): Active Database Systems. In: Kim, W. (Hrsg.): *Modern Database Systems*, ACM Press, New York, NJ, 1995, S. 434-456.

Eder, J./Liebhart, W. (1996): Workflow Recovery. In: *Proc. of the 1st IFCIS Int. Conf. on Cooperative Information Systems, CoopIS'96*, IEEE Computer Society Press, Los Alamitos, CA, 1996, S. 124-134.

Elmagarmid, A.K. (1992): *Database Transaction Models For Advanced Applications*. Morgan Kaufmann Publishers, San Mateo, CA, 1992.

Georgakopoulos, D./Hornick, M./Sheth, A. (1995): An Overview of Workflow Management: From Process Modeling to Workflow Automation Infrastructure. *Distributed and Parallel Databases*, 3(2), 1995, S. 119-153.

Jablonski, S./Bussler, C. (1996): *Workflow Management - Modeling Concepts, Architecture and Implementation*. International Thomson Publishing, London, 1996.

Korth, H./Levy, E./Silberschatz, A. (1990): A Formal Approach to Recovery by Compensating Transactions. In: McLeod, D. et al. (Hrsg.): *Proc. of the 16th Int. Conf. on Very Large Data Bases (VLDB'90)*, Morgan Kaufmann Publishers, Palo Alto, CA, 1990, S. 95-106.

Leymann, F. (1996): Transaktionskonzepte für Workflow-Management-Systeme. In: Vossen, G. et al. (Hrsg.): *Geschäftsprozeßmodellierung und Workflow-Management: Modelle, Methoden, Werkzeuge*, Kapitel 19. International Thomson Publishing, 1996.

Moss, J.E.B. (1985): *Nested Transactions: An Approach to Reliable Distributed Computing*. MIT Press, Cambridge, MA, 1985.

Rusinkiewicz, M./Klas, W./Tesch, T./Wäsch, J./Muth, P. (1995): Towards a Cooperative Transaction Model - The Cooperative Activity Model. In: Dayal, U. et al. (Hrsg.): *Proc. of the 21st Int. Conf. on Very Large Data Bases (VLDB'95)*, Morgan Kaufmann Publishers, 1995, S. 194-205.

Reuter, A./Schneider, K./Schwenkreis, F. (1997): ConTracts Revisited. In: Jajodia, S. et al. (Hrsg.): *Advanced Transaction Models and Architectures*, Kapitel 5, Kluwer Academic Publishers, Boston, 1997, S. 127-151.

Sheth, A./Rusinkiewicz, M. (1993): On Transactional Workflows. *Bulletin of the IEEE Technical Committee on Data Engineering*, 16(2), 1993, S. 37-40.

Schwarz, K./Türker, C./Saake, G. (1998a): Execution Dependencies in Transaction Closures. In: Halper, M. (Hrsg.): *Proc. of the 3rd Int. Conf. on Cooperative Information Systems, CoopIS'98*, IEEE Computer Society Press, Los Alamitos, CA, 1998, S. 122-131.

Schwarz, K./Türker, C./Saake, G. (1998b): Extending Transaction Closures by N-ary Termination Dependencies. In: Litwin, W. et al. (Hrsg): *Second East-European Symposium on Advances in Databases and Information Systems (ADBIS'98)*, Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1998, S. 131-142.

Schwarz, K./Türker, C./Saake, G. (1998c): Transitive Dependencies in Transaction Closures. In: Eaglestone, B. et al. (Hrsg.): *Proc. of the 1998 Int. Database Engineering and Applications Symposium (IDEAS'98)*, IEEE Computer Science Press, Los Alamitos, CA, 1998, S. 34-43.

Worah, D./Sheth, A. (1997): Transactions in Transactional Workflows. In: Jajodia, S und Kerschberg, L. (Hrsg.): *Advanced Transaction Models and Architectures*, Kluwer Academic Publishers, Boston, 1997, S. 3-34.