# Interoperabilität heterogener Workflows

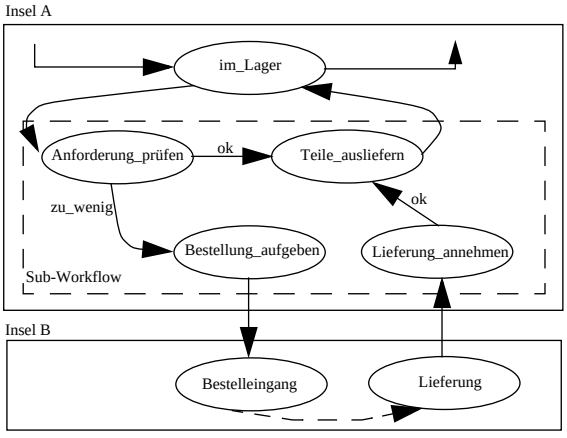
Markus Bon, Norbert Ritter, Jürgen Zimmermann
Universität Kaiserslautern
67653 Kaiserslautern
{bon|ritter|jnzimmer}@informatik.uni-kl.de

**Abstract:** *Workflow-Management-Systeme* haben sich als sinnvolle und wichtige Hilfsmittel zur Überwachung und Organisation betrieblicher Prozesse etabliert. Leider gibt es jedoch eine große Anzahl verschiedener Systeme, die den Begriff des Workflows auch vollkommen verschieden interpretieren und umsetzen. Trotz der Bemühungen der WfMC auf diesem Gebiet ist die Standardisierung von Workflows noch in weiter Ferne. Nichtsdestotrotz ist es in größeren Firmenverbünden notwendig, daß diese völlig verschiedenen Systeme zusammenarbeiten. Im diesem Artikel werden erste Ideen präsentiert, wie Workflow-Interoperabilität erreicht werden kann. Es werden Mechanismen vorgestellt, die es erlauben, einen globalen Workflow aus lokalen Workflows zusammenzusetzen. Zusätzlich wird die Gestaltung einer entsprechenden Ausführungsumgebung angedacht.

## 1. Motivation

Moderne Unternehmen, insbesondere große Konzerne, bestehen selten aus einem einzigen Betrieb, sondern vielmehr aus mehreren Teilunternehmen. Die Zusammensetzung ändert sich häufig, beispielsweise durch Hinzukauf bzw. Verkauf von Unternehmensteilen. Es können aber auch losere Beziehungen zu anderen Unternehmen bestehen, z. B. zu Lieferanten. Da jedes Teilunternehmen in der Regel über eine eigene EDV verfügt, können die eingesetzten Software-Systeme völlig verschieden sein. Da unser Ziel die Modellierung und rechnergestützte Steuerung von Geschäftsprozessen ist, die mehrere Teilunternehmen betreffen, behandeln wir die daraus resultierenden Fragen der Heterogenität und Interoperabilität. Einen Geschäftsprozeß, der sich über mehrere (Teil-)Unternehmen bzw. in diesen eingesetzten, heterogenen Workflow-Management-Systeme (WfMS) erstreckt, bezeichnen wir als **heterogenen Workflow**. Beispiel 1 zeigt einen solchen Geschäftsprozeß. Auf Seite der Lagerverwaltung (A) beschreibt ein Workflow, wie im Lager auf eintreffende Anforderungen zu reagieren ist. Wird ein Mangel an Teilen festgestellt, so wird eine Bestellung an einen geeigneten Lieferanten (B) aufgegeben. Zur Bearbeitung dieser Bestellung wird auch auf der Seite des Lieferanten ein entsprechender Workflow initiiert. Da Lagerverwaltung und Lieferant aber in der Regel verschiedene WfMS einsetzen, muß eine geeignete Art der Verständigung zwischen diesen **Workflow-Inseln** (A, B) gefunden werden. Eine Workflow-Insel besteht dabei aus einem WfMS (mit zugehörigem Modell), einer Menge von Ressourcen (Applikationen und/oder organisatorische Einheiten) und ggf. einer Ansammlung bereits definierter Schemata.

Beispiel 1 Anwendungsbeispiel: Lagerverwaltung

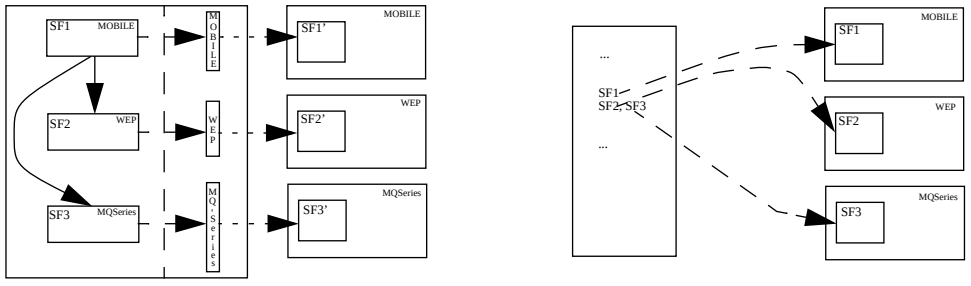


Innerhalb der Workflow-Inseln werden ausschließlich lokale Aspekte betrachtet. Dabei handelt es sich jeweils um den Teil des Gesamtprozesses, der in der entsprechenden Insel abgearbeitet werden soll und kann. Folglich muß die Koordination der lokalen (Teil-)Workflows auf einer höheren Ebene stattfinden. Zwei Vorgehensweisen sind denkbar, um die für diese übergeordnete Koordination notwendigen Mechanismen zu entwickeln: bei **Bottom-Up** werden bereits existierende lokale Workflow-Schemata genutzt, um daraus das Gesamtschema zusammenzusetzen; bei **Top-Down** hingegen wird zunächst im Wissen um die Gesamtmenge der auf den einzelnen Inseln zur Verfügung stehenden Resourcen der Gesamtprozeß (in einer einheitlichen Sprache) beschrieben, anschließend werden die durch die einzelnen Inseln ausführbaren Teile (automatisch oder manuell) extrahiert und an diese delegiert.

### 2. Übergeordnete Koordinatorebene

Ungeachtet der gewählten Vorgehensweise muß auf einer den Inseln übergeordneten Ebene der Datenfluß zwischen den Inseln und der Kontrollfluß des Gesamt-Workflows beschrieben sein. Da hier die Koordination zwischen den Inseln geschieht, bezeichnen wir diese Ebene als **Koordinatorebene.** Zur Definition des Kontrollflusses können die üblichen Kontrollstrukturen wie Sequenz, Parallelausführung, bedingte Verzweigung oder Iteration eingesetzt werden. Jeder Schritt im Kontrollfluß stellt einen potentiellen Übergang von einer Insel zur nächsten dar. Der auf eine Insel verlagerte Sub-Workflow muß in eine dem lokalen System verständliche Form gebracht und von der entsprechenden WF-Engine ausgeführt werden. Um dies zu erreichen sind zwei Beschreibungsverfahren denkbar, die wir als **Delegation,** und **Integration** bezeichnen. In Abbildung 1 sind beide Verfahren gezeigt: bei der Delegation (a) liegen die Beschreibungen der lokalen Workflows vollständig auf der Koordinatorebene vor. Sie werden durch „Mapper“ in die Sprache des lokalen WfMS abgebildet. Dadurch werden auf der Insel neue Workflows erzeugt, die anschließend lokal verarbeitet werden. Die Delegation unterstützt somit Top-Down-Entwürfe. Bei der Integration (b) hingegen werden nur schon existierende Workflows benutzt. Die Sub-Workflow-Definition liegt bereits auf dem Zielsystem vor, sie wird in die übergeordnete Beschreibung nur über eine entsprechende Schnittstelle integriert. Dies entspricht der Bottom-Up-Vorgehensweise.

Abbildung 1 Beschreibungsverfahren für übergreifende Sub-Workflows




**a) Delegation** **b) Integration**

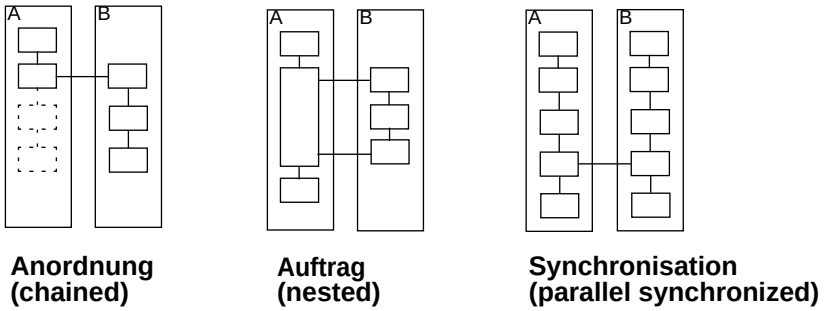
In den meisten Fällen läßt sich jedoch kein strikter „Bottom-Up“ oder „Top Down“ Ansatz durchführen, es wird vielmehr zu einer Vermischung kommen: Während die lokalen Modelle meist schon existieren und eingesetzt werden, entstehen die globalen Abläufe in den Köpfen der Ablaufmodellierer neu. Es ist somit eine Annäherung aus beiden Richtungen notwendig!

### 2.1 Abhängigkeiten

In Abbildung 2 sind die in [WfMC96] beschriebenen möglichen Abhängigkeiten zwischen interoperierenden Workflows dargestellt. Bei der **Anordnung** wird aus einem auf Insel A initiierten Workflow heraus ein Workflow auf Insel B initiiert. Beim **Auftrag** wird ähnlich wie bei der Anordnung auf Insel B ein neuer Workflow erzeugt und zur Ausführung gebracht, jedoch wartet der A-Workflow solange, bis der B-Workflow seine Arbeit beendet hat und kann dann den durch den B-Workflow erreichten Zustand in seiner eigenen, weiteren Verarbeitung berücksichtigen. Das Szenario aus Beispiel 1 fällt in diese Kategorie. Beide Abhängigkeiten sind aufgrund des „Block“-Charakters der Interaktion als eher unkritisch

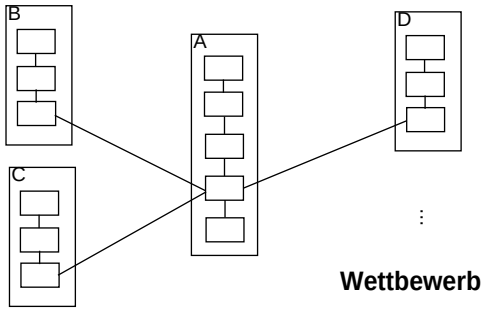
einzustufen. Bei der **Synchronisation** sieht das allerdings anders aus, denn die beiden Workflows werden i. allg. unabhängig voneinander initiiert. An ausgezeichneten Synchronisationspunkten wird die Ausführung des „Schnelleren“ gestoppt, bis der zugehörige Synchronisationspartner ebenfalls den Synchronisationspunkt erreicht hat. Nach einem entsprechenden Abgleich der erreichten Zustände können beide Workflows weiter ausgeführt werden. Die Anzahl der Synchronisationspunkte ist nicht beschränkt. Daher kann es zu einer durchaus regen Kommunikation zwischen den Inseln kommen.

Abbildung 2 Abhängigkeiten




Neben diesen drei Abhängigkeiten sind jedoch noch weitere Szenarien denkbar. Von der WfMC nicht abgedeckt wird z. B. eine Wettbewerbssituation, wie sie in Abbildung 3 zu sehen ist. Hier kann es sich beispielsweise um das Einholen von Angeboten handeln. Die Anbieter (mit den zugehörigen Inseln B, C, D, ...) dürfen für eine vereinbarte Zeitspanne Angebote übermitteln, jedoch ist im Vorfeld die Anzahl der teilnehmenden Inseln nicht bekannt.

Abbildung 3 Abhängigkeit mit im voraus unbekannter Inselzahl




## 2.2 Aufgaben der Koordination

Durch die Verteilung des Gesamtablaufs auf viele lokale Systeme muß für eine entsprechende Koordination des Ablaufs gesorgt werden. Diese umfaßt:

- *Interpretation des globalen Workflow-Schemas;*
  Ziel des heterogenen WfMS ist die Durchführung einer Gesamtaufgabe mit Hilfe vieler, räumlich verteilter Teilworkflows, die entsprechend des globalen Schemas bestimmt werden.
- *Initiieren der erforderlichen Workflow-Exemplare auf den Inseln;*
  Wurde ein Teil-Workflow zur Ausführung einer Teilaufgabe ausgewählt, so muß auf der zugehörigen Insel ein Workflow-Exemplar erzeugt und mit Daten versehen werden.
- *Kontrolle der Abhängigkeiten zwischen den lokalen Workflows;*
  Die Kommunikation zwischen den Inseln muß gewährleistet werden. Bei der Synchronisation müssen beispielsweise zusammengehörige Synchronisationspunkte überwacht werden.
- *Herstellung der Kommunikationsverbindungen zwischen den interoperierenden, lokalen Workflows;*
  Zum Datenaustausch zwischen Inseln müssen die zugehörigen Kommunikationspartner lokalisiert und entsprechende Nachrichten ausgetauscht werden.

- *Ableiten eines globalen Zustands durch Abfrage der lokalen Zustände;*
  Um ein Bild vom aktuellen Stand der Verarbeitung zu bekommen, müssen die Zustände aller (aktiven) Teilabläufe gesammelt und bewertet werden.
- *Unterstützung von „Monitoring“ und weiterer Administrationsfunktionalität;*
  „Monitoring“ ermöglicht das Überwachen eines Ablaufs zum Ausführungszeitpunkt. Falls ein Eingriff von qualifizierter Seite notwendig wird (Fehlerfall), soll dies durch geeignete Werkzeuge ermöglicht werden.
- *Protokollierung des Ablaufs („History“).*
  Zu einer nachträglichen Analyse (z. B. zur Fehlersuche, Optimierung oder Qualitätskontrolle) müssen genug Informationen protokolliert werden, um die Durchführung des Workflows nachvollziehbar zu machen.

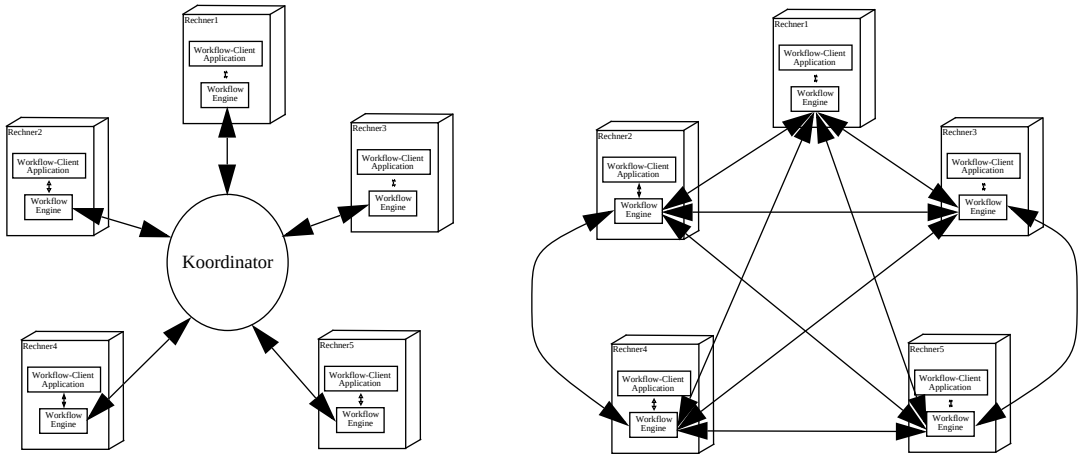
### 2.3 Architekturansätze

Zur Durchführung der vorgestellten Aufgaben sind verschiedene Ansätze denkbar. Zwei mögliche Architekturansätze werden im folgenden Abschnitt vorgestellt. Im zentralen Fall übernimmt eine spezielle Komponente die Koordination, im verteilten Fall wird diese Aufgabe von den Inseln übernommen.

#### 2.3.1 Zentrale Kontrolle

Auf der Koordinatorebene liegt eine vollständige Beschreibung des globalen Workflows vor. Zur Kontrolle der Ausführung dient eine zentrale Komponente, die dieses globale Schema kennt. Diese Komponente bezeichnen wir als Koordinator. Dem Koordinator ist nicht nur die Beschreibung des Gesamtprozesses zugänglich, auch die Zielsysteme, auf denen die entsprechenden Sub-Workflows ausgeführt werden sollen, sind ihm bekannt. Weiterhin läuft auch die Kommunikation zwischen den Inseln über den Koordinator. Aufgrund dieser vollständigen Information sind die zuvor geforderten Aufgaben recht einfach lösbar, insbesondere die Überwachung des Gesamtzustands und das Mitprotokollieren des Ablaufs.
Die Abwicklung der Kommunikation über den Koordinator stellt Anforderungen sowohl an den Koordinator selbst, als auch an die Inseln: Der Koordinator muß den Inseln eine API anbieten, die das Versenden von Nachrichten ermöglicht. Durch die Kenntnis des globalen Schemas ist es ihm möglich, selbstständig die Zielinsel festzustellen und die Nachricht auszuliefern. Weiterhin kann eine so verschickte Nachricht protokolliert werden.
Auf der Inselseite benötigen wir zur Zusammenarbeit mit dem Koordinator eine API, die ihm die Auslieferung einer Nachricht an die Insel gestattet. Darüber hinaus muß die Insel-API das Erzeugen und Initiieren von Workflows, sowie die Abfrage des lokalen Zustands durch den Koordinator unterstützen.

Abbildung 4 Kontrolle der Ausführung




**a) zentrale Kontrolle** **b) voll verteilt - dezentrale Kontrolle**

Durch den Einsatz einer zentralen Komponente kann natürlich ein Flaschenhals mit den schon bekannten Problemen entstehen. Da der Koordinator jedoch durchaus verteilt umgesetzt werden kann, kann auch diese Problematik entschärft werden.

### 2.3.2 Dezentrale Kontrolle

Eine andere Idee besteht darin, den Gesamt-Workflow verteilt zu realisieren (Abbildung 4b) [Schu99]. Der jeweils aktive Knoten trifft die Entscheidung, wie weiter zu verfahren ist, und gibt die Kontrolle entsprechend weiter. Hierbei muß jedoch nicht nur ein neuer Workflow auf der Insel erzeugt werden, vielmehr müssen auch alle für den weiteren Ablauf nötigen Informationen weitergegeben werden. Es ergeben sich jedoch nicht zu unterschätzende Probleme: Jedes auf einer Insel befindliche WfMS muß diese Art der Kommunikation von Workflow-Zuständen unterstützen, was eine gravierende Erweiterung der bestehenden Systeme erfordert. Das Lokalisieren der Insel, die als nächste die Kontrolle übernimmt, ist ebenfalls problematisch: durch das Fehlen einer zentralen Instanz ist der Einsatz teurer Mechanismen wie „Broadcast“ notwendig. Das Feststellen eines globalen Zustands und „Monitoring“ wird durch die Migration von Insel zu Insel, bzw. der gleichzeitigen Aktivierung mehrerer Inseln bei Verzweigung des Ablaufs, geradezu unmöglich. Auch die Wünsche, eine Art „Historie“ in Form eines Protokolls zu führen, bzw. im Fehlerfall in einem früheren Schritt wieder aufzusetzen, sind durch das Fehlen einer zentralen Überwachung nahezu unerfüllbar. Eine Insel beendet nach Erfüllen ihres Teilablaufs ihre Aktivität mit Weitergabe der Kontrolle (ansonsten wird ein aufwendiges Beenden der verteilten Workflows notwendig). Insbesondere steht sie dann auch nicht mehr zur Auskunft über den lokalen Zustand oder die bisherige Vorgeschichte zur Verfügung.

## 3. Zusammenfassung und Ausblick

Wir haben die Notwendigkeit der Interoperation heterogener Workflows motiviert und erste Ideen präsentiert, wie heterogene Workflows, die auf Workflow-Inseln verteilt sind, zu einem übergeordneten Schema vereinigt werden können. Wir haben dabei zum einen die Top-Down-Vorgehensweise betrachtet, bei der eine vorliegende Beschreibung des globalen Ablaufs partitioniert und auf die Inseln verteilt wird, zum anderen den Bottom-Up-Ansatz, mit dessen Hilfe existierende Workflows integriert werden. Weiterhin wurden die von der WfMC identifizierten Abhängigkeiten zwischen interoperierenden Workflows vorgestellt und an einem Beispiel das Vorhandensein weiterer Abhängigkeiten angedeutet.
Abschließend wurde die Ausführungsumgebung betrachtet und die zentrale Kontrolle der Ausführung durch einen Koordinator der dezentralen gegenübergestellt. Hierbei ergab sich die Einschätzung, das eine voll verteilte Lösung zu viele schwer lösbare Probleme mit sich bringt und dem Koordinator-Ansatz der Vorzug zu geben ist.
Unsere Aufgabe für die Zukunft ist nun die Umsetzung der hier angedachten Ideen und das Überprüfen ihrer Eignung in der wirklichen Welt. Hierzu sollen konkrete Anwendungsbeispiele eines industriellen Kooperationspartners untersucht und sich daraus ergebende Integrationsansätze erarbeitet werden. Weiterhin steht die Entwicklung und (zumindest prototypische) Implementierung einer Ablaufkomponente auf dem Plan, die angesichts der in Abschnitt 2.3 vorgebrachten Argumente den Ansatz der zentralen Kontrolle verfolgen wird. Entwickelt werden müssen hierzu der Koordinator an sich (mit geeigneter API) und eine Möglichkeit der Erweiterung vorhandener WfMS um eine Inselseitige API.

## 4. Referenzen


[Schu99] Wolfgang Schulze, *Workflow-Management für CORBA-basierte Anwendungen*, Springer-Verlag, Berlin Heidelberg 2000

[WfMC96] Workflow Management Coalition, *Workflow Standard - Interoperability*, October 1996, Document Number WfMC TC-1012, www.aiim.org/wfmc/mainframe.htm

[WfMC98] Workflow Management Coalition, Work Group 1, *Interface 1: Process Definition Interchange — Process Model*, November 1998, Document Number WfMC TC-1016-P, www.aiim.org/wfmc/mainframe.htm