Interoperable Workflows

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

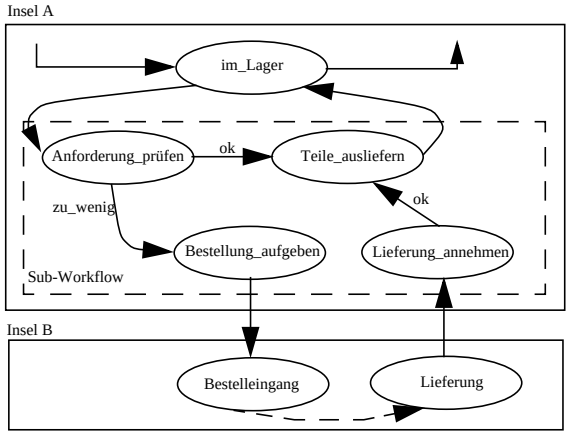
1. Motivation

Entwicklung und Einsatz von Workflow-Management-Systemen (WfMS) sind in den letzten Jahren zu sehr wichtigen Themen in der Informatik herangewachsen. Grundidee dabei ist, Geschäftsprozesse möglichst exakt zu beschreiben, um rechnergesteuert einzelne Aufgaben so zu verteilen, daß die richtige Person zum richtigen Zeitpunkt alles Notwendige tun kann. Auf diese Art und Weise können auch komplexere Abläufe optimiert, Ressourcen besser ausgenutzt und unnötige Verzögerungen vermieden werden. Dies spart Zeit und Geld!
Leider gibt es für WfMS immer noch einen ziemlichen Wildwuchs an Ideen und Konzepten. Es gibt dementsprechend auch eine große Anzahl verschiedener Systeme, die leider alles andere als kompatibel zueinander sind. Hinzu kommen Systeme, die für sehr spezielle Bereiche wie z. B. Ingenieur-Prozesse entwickelt wurden [BDS98].
Darüber hinaus kann die Struktur moderner Unternehmen sehr komplex sein. Oft sind viele Teilunternehmen beteiligt. Jederzeit können auch neue Unternehmen hinzugekauft oder wieder abgestoßen werden. Weiterhin gibt es noch Beziehungen zu externen Unternehmen, beispielsweise Zulieferern. Jedes dieser Unternehmen kann eine eigene IT-Abteilung besitzen und vollkommen verschiedene Systeme einsetzen. Nichtsdestotrotz besteht natürlich der Wunsch, unternehmensweit oder sogar unternehmensübergreifend Prozesse zu beschreiben, und mittels eines geeigneten WfMS zu koordinieren. Genau dieses Ziel wollen wir erreichen.
Im folgenden Artikel werden wir die auftretenden Probleme der Heterogenität und Interoperabilität näher untersuchen und erste Ideen präsentieren, wie globale Workflows beschrieben werden können. Weiterhin werden verschiedene Grade der Abhängigkeit zwischen den in den einzelnen Firmen ausgeführten Teilworkflows untersucht. Eine geeignete Architektur wird ebenfalls andiskutiert.

2. Begriffe

Zunächst sollen einige Begriffe eingeführt werden, die für die weiteren Ausführungen benötigt werden.

Beispiel 1 Anwendungsbeispiel: Lagerverwaltung



2.1 Heterogener Workflow

Unter einem *heterogenen Workflow* verstehen wir die Umsetzung eines Geschäftsprozesses auf hetero-

genen Systemen Die Durchführung des Prozesses betrifft mehrere (Teil-)Unternehmen und damit in der Regel auch verschiedene WfMS. In *Beispiel 1* ist ein Ausschnitt aus einer Lagerverwaltung zu sehen. Fehlen im Lager Teile, müssen sie bei einem externen Lieferanten bestellt werden, der diese anschließend liefert. Somit müssen zwei verschiedene WfMS – in unserem Beispiel eins auf Lager- und eins auf Lieferantenseite – miteinander in Einklang gebracht werden.

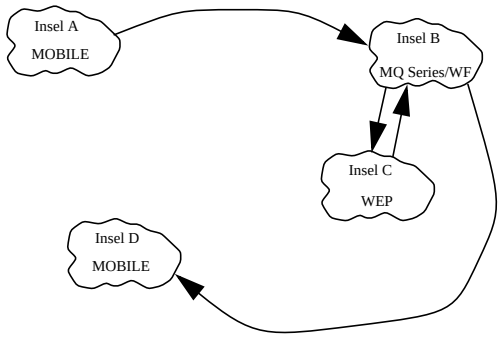
2.2 Workflow-Insel

Eine *Workflow-Insel* bezeichnet die Einheit eines WfMS (als Implementierung eines zugehörigen Workflow-Modells), einer Menge von Ressourcen (Applikationen und/oder organisatorische Einheiten) und ggf. einer Menge bereits definierter (lokaler) Workflow-Schemata.

2.3 Globaler Workflow

Zwischen den Inseln können Abhängigkeiten bestehen, wenn Abläufe über Inselgrenzen hinweg miteinander kooperieren sollen. In Abbildung 1 sind zwei verschiedene Abhängigkeiten angedeutet. Ein Workflow auf Insel A stößt einen weiteren Workflow auf Insel B an. Mit dem Start des B-Workflows endet die Kommunikation (analog sieht es mit den Inseln B und D aus). Dieser Workflow gibt einen Auftrag an Insel C und wartet anschließend auf Rückantwort. Erst dann setzt er seine Verarbeitung fort. Auf die verschiedenen Formen dieser Abhängigkeiten werden wir Abschnitt 3 noch zu sprechen kommen. Die Gesamtheit aller beteiligten Insel-Workflow-Instanzen zusammen mit dem dazugehörigen Kontrollfluß und der Beschreibung ihrer Abhängigkeiten bezeichnen wir als *globalen Workflow*.

Abbildung 1 Inseln mit Abhängigkeiten



2.4 Globales Workflow-Schema

Das *globale Workflow-Schema* dient der Beschreibung von globalen Workflows eines Typs. Der Umfang des Schemas hängt eng mit der gewählten Strategie zusammen; mögliche Strategien werden in Abschnitt 2.6 vorgestellt.

2.5 Koordinator-Ebene

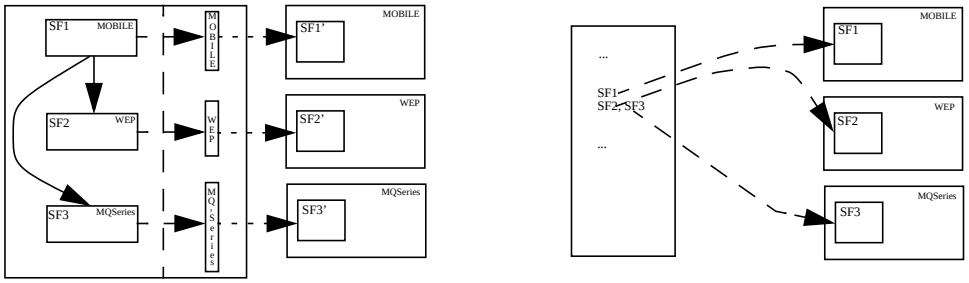
Bei heterogenen Workflows sind die beteiligten Sub-Workflows über viele Inseln verteilt. Daher benötigen wir eine übergeordnete Sicht, die Lokalisieren und Koordination der einzelnen Sub-Workflows ermöglicht. Auch müssen auf dieser Ebene die Kommunikation zwischen den Inseln und die Überwachung des Gesamtablaufs stattfinden. Auf Grund dieser Aufgaben haben wir diese Schicht *Koordinator-Ebene* getauft.

2.6 Delegation vs. Integration

Zur Umsetzung eines globalen Workflows gibt es ein breites Spektrum an Möglichkeiten. Zwei davon sind *Delegation* und *Integration*. Bei der Delegation wird ein Top-Down-Entwurf umgesetzt: das komplette globale Workflow-Schema wird in einer einzigen Definitionssprache (WfDS) spezifiziert. Anschließend werden Sub-Schemata identifiziert und mit Hilfe spezieller Mapper in eine Form übersetzt, in der sie von den auf den Ziel-Inseln angesiedelten WfMS verarbeitet werden können (Abbildung 2a).

Im Gegensatz dazu wird bei der Integration „buttom up“ vorgegangen. Ausgehend von schon existierenden Ablaufbeschreibungen in den Zielsystemen wird ein globales Workflow-Schema definiert, das nur noch schon vorhandene Sub-Schemata einbindet(Abbildung 2b).
Beide Vorgehensweisen werden wohl nie in Reinform auftreten. Da viele der benutzten Sub-Schemata schon vorliegen und auch weiter genutzt werden sollen, steht hier die Integration im Vordergrund. Bei der Spezifikation des globalen Ablaufs, die üblicherweise vom Management durchgeführt wird, soll aber nicht ausschließlich das Aufrufen schon existierender Schemata zur Verfügung stehen, auch die Möglichkeit der Modellierung komplexerer Abhängigkeiten ist wünschenswert. Hier greift eher die Delegation.

Abbildung 2 Beschreibungsverfahren für übergreifende Sub-Workflows

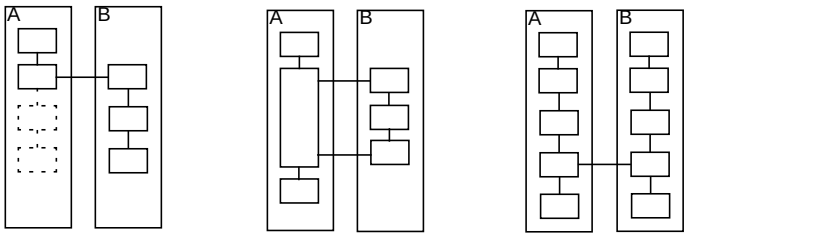


a) Delegation **b) Integration**

3. Abhängigkeiten

In [WfMC96] werden verschiedene Grade der Abhängigkeit für interoperierende Workflows beschrieben. Es handelt sich dabei um Anordnung („chained“), Auftrag („nested“) und Synchronisation („parallel synchronized“). Das einfachste Szenario bietet dabei die *Anordnung*: Ein Workflow-Exemplar, das auf Insel A ausgeführt wird, initiiert die Ausführung eines Sub-Workflows auf Insel B. Der weitere Ablauf des neu erzeugten Exemplars spielt für den „A-Workflow“ keine Rolle mehr, es findet keine weitere Kommunikation statt.
Etwas komplexer ist der *Auftrag*. Nachdem der Workflow auf Insel A den Sub-Workflow auf B gestartet hat, wartet er, bis dieser abgearbeitet wurde. Danach wird in irgendeiner Form ein Ergebnis an Insel A zurückgeliefert, der dortige Workflow läuft weiter.

Abbildung 3 Abhängigkeiten

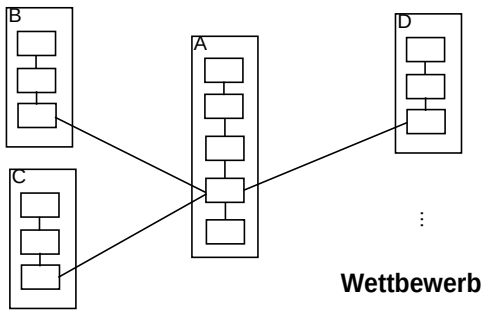


Anordnung (chained) **Auftrag (nested)** **Synchronisation (parallel synchronized)**

Beide Szenarien sind insofern „einfach“, daß der gewünschte Sub-Workflow als „Black Box“ ausgeführt werden kann. Bei der Synchronisation ist das nicht ganz so leicht. Beide Workflow-Exemplare befinden sich bereits in der Ausführung. An ausgezeichneten Stellen findet jedoch eine Kommunikation miteinander statt, die zur reinen Synchronisation, aber auch zum Austausch von Daten genutzt werden kann.
Der Workflow, der zuerst seinen Synchronisationspunkt erreicht, wartet, bis auch sein Partner-Workflow kommunikationsbereit ist. Nach der Abstimmung mit evtl. Datenaustausch laufen beide Workflows weiter. Synchronisation zwischen zwei Workflows kann beliebig oft stattfinden, es kann also durchaus zu regem Kontakt zwischen den Inseln kommen. In Abbildung 3 sind die beschriebenen Abhängigkeiten dargestellt.

Von der WfMC nicht behandelt, aber durchaus sinnvoll zu betrachten, sind Szenarien mit im voraus unbekannter Anzahl teilnehmender Inseln. In Abbildung 4 ist z. B. der *Wettbewerb* zu sehen. Hierbei kann es sich beispielsweise um eine Ausschreibung handeln. Die interessierten Firmen haben lokal Workflows zur Angebotserstellung laufen. Der Workflow auf Insel A muß die eingehenden Angebote sammeln und verarbeiten. Die „Bewerbungsphase“ ist zeitlich begrenzt, die Anzahl der Bewerber ist im voraus völlig unklar.

Abbildung 4 Abhängigkeit mit im voraus unbekannter Inselanzahl



4. Der Koordinator

4.1 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 Teil-Workflows, die entsprechend des globalen Schemas bestimmt werden. Bei der Integration geschieht dies durch Einbinden geeigneter Teil-Workflows, die auf verschiedenen Insel schon bereitstehen. Bei der Delegation werden Teile des globalen Schemas in entsprechenden Einheiten auf geeignete Inseln verteilt.
- *Initiieren der erforderlichen Workflow-Exemplare auf den Inseln*
  Der Kontrollfluß wird durch den globalen Workflow bestimmt. Wird ein Teil-Workflow durch den Kontrollfluß zur Ausführung einer Teilaufgabe ausgewählt, so muß die zugehörige Insel bestimmt werden. Dort wird ein Workflow-Exemplar erzeugt, mit Daten versehen und gestartet.
- *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 übermittelt 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 Informationen protokolliert werden, so daß die Durchführung des Workflows nachvollziehbar ist.

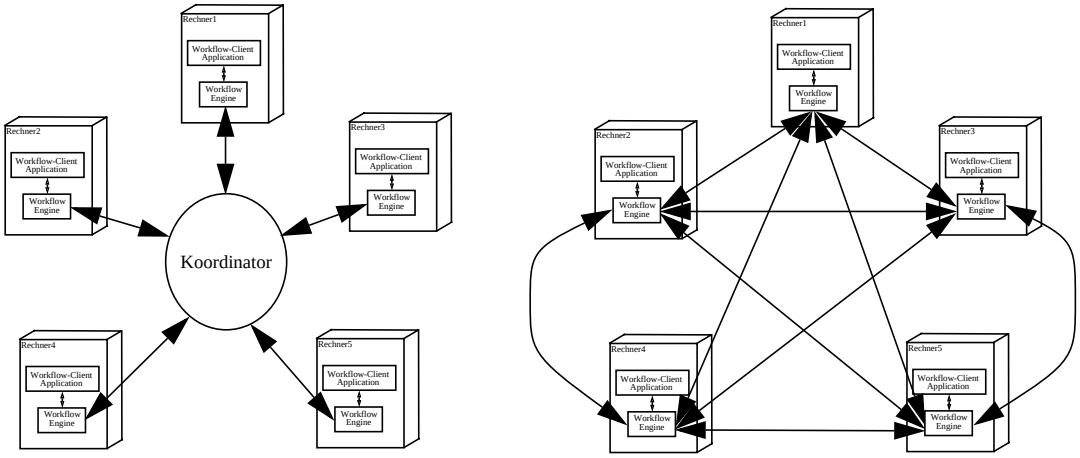
4.2 Zentral vs. dezentrale Koordination

Um die im vorigen Abschnitt aufgezählten Aufgaben zu bewältigen, sind zwei Extremformen der Organisation denkbar: zum einen eine die Einführung einer zentralen Kontrollinstanz, zum anderen eine völlige Verteilung der Kontrolle über alle teilnehmenden Knoten.

Beide Ansätze haben ihre Stärken und Schwächen, die hier kurz diskutiert werden sollen. Beim zentralen Ansatz existiert eine Koordinator-Komponente (die wir auch kurz als *Koordinator* bezeichnen). In dieser Komponente ist die Beschreibung des globalen Workflows abgelegt. Alle verfügbaren Sub-Workflows mit zugehöriger Insel werden hier registriert und können bei Bedarf vom Koordinator lokalisiert werden. Weiterhin wird die Kommunikation zwischen den Inseln vom Koordinator übernommen. Eine Analyse des Nachrichtenverkehrs erleichtert das Monitoring, da flexibel auf eintretende Ereignisse reagiert werden kann; auch Mitprotokollieren des Nachrichtenverkehrs zur nachträglichen Analyse ist sehr einfach möglich. Somit ergibt sich für den Koordinator (vorerst) die in Abbildung 6 dargestellte Komponentenaufteilung.

Die Abwicklung der Kommunikation durch den Koordinator stellt natürlich Anforderungen sowohl an den Koordinator selbst, als auch an die Inseln. Auf Koordinatorseite muß den Inseln ein API zur Verfügung stehen, das die gemäß der bestehenden Abhängigkeiten notwendige Kommunikation ermöglicht. Durch die Kenntnis des Gesamtschemas und aller bereitstehenden Teil-Workflows ist ein Lokalisieren der Insel eines Ziel-Workflows sehr einfach möglich. Außerdem erleichtert sich das Ermitteln eines Gesamtzustands durch Abfrage der lokalen Zustände aller beteiligten Teil-Workflows.

Abbildung 5 Kontrolle der Ausführung



a) zentrale Kontrolle **b) dezentrale Kontrolle**

Inselseitig muß dem Koordinator ermöglicht werden, neue Workflow-Exemplare zu erzeugen und zu starten, bzw. bei der Synchronisation Daten auszuliefern. Weiterhin muß die Insel bei Anfrage Auskunft über den lokalen Zustand eines Workflow-Exemplars geben können.

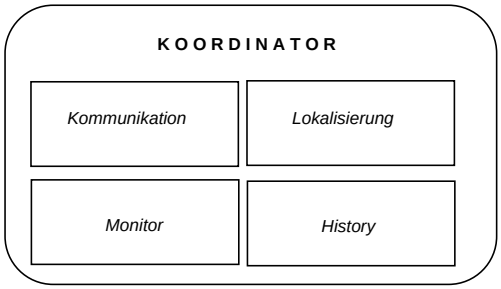
Natürlich hat der zentrale Ansatz auch Nachteile: Es müssen Vorkehrungen getroffen werden, um den Koordinator nicht zum „single point of failure“ werden zu lassen. Ein Ausfall dieser Schaltzentrale hätte katastrophale Folgen! Weiterhin ist eine verteilte Implementierung des Koordinators vorzusehen, da sonst bei dem zu erwartenden Kommunikationsaufkommen leicht Engpässe entstehen können.

Bei dem verteilten Ansatz fehlt die zentrale Komponente. Die Kontrolle wird an die jeweils aktiven Inseln weitergegeben. Nachdem ein aktiver Knoten entschieden hat, wie weiter zu verfahren ist, erzeugt er auf der Zielinsel ein entsprechendes Workflow-Exemplar. Anschließend werden alle weiteren zum Ablauf notwendigen Informationen (insbesondere das globale Schema) übertragen. Das System ist gut skalierbar, und es sind wenig Engpässe zu erwarten. Die Idee klingt sehr elegant, wirft aber eine ganze Menge von Problemen auf: die momentan verfügbaren WfMS unterstützen in der Regel diese Art von Migration nicht. Folglich wären grundlegende Erweiterungen notwendig. Das Auffinden der Inseln muß

fest kodiert sein, ansonsten werden teure Broadcast-Mechanismen notwendig. Das Feststellen eines globalen Zustands gestaltet sich durch die Verteilung sehr schwierig. Monitoring ist nur durch Einführen einer zentralen Monitoring-Komponente möglich [HLKP2000].

Aus diesen Gründen haben wir beschlossen, den verteilten Ansatz nicht weiter zu verfolgen und uns der Koordinator-Idee zuzuwenden.

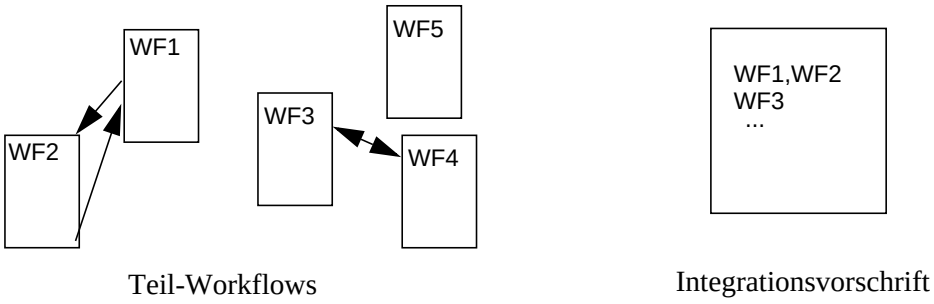
Abbildung 6 Komponenten des Koordinators



5. Integration heterogener Sub-Workflow-Schemata

Wie sieht bei der Integration von (bestehenden) Workflow-Schemata nun die genaue Vorgehensweise aus? Die Ausgangssituation stellt sich wie in Abbildung 7 gezeigt dar: Es existiert eine Menge schon definierter Workflow-Schemata (WF1 bis WF5), zwischen denen auch Abhängigkeiten bestehen können (WF1 gibt einen Auftrag an WF2, WF3 synchronisiert sich mit WF4). Ferner existiert eine Beschreibung des globalen Workflows (*Integrationsvorschrift*).

Abbildung 7 Ausgangssituation



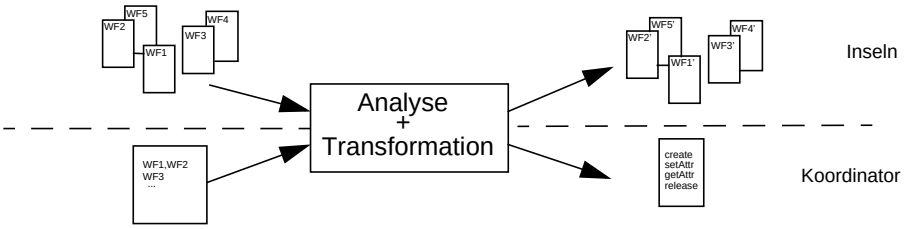
Unser Ziel ist es nun, daraus einen ablauffähigen, globalen Workflow zu formen, der mit Hilfe der Koordinatorkomponente ausgeführt werden kann. Dazu sind einerseits Anpassungen an den Teil-Workflows notwendig, um die Zusammenarbeit mit dem Koordinator zu ermöglichen. Andererseits muß der Koordinator natürlich mit allen Informationen versorgt werden, die er zur Abwicklung des globalen Ablaufs benötigt. Zu diesem Zweck brauchen wir ein Werkzeug, das als Eingabe die Schemata der Teil-Workflows und die Integrationsvorschrift nutzt, um daraus die modifizierten Workflow-Schemata für die Inseln und den Koordinator zu generieren (Abbildung 8). Auf diese Weise erfährt der Koordinator, auf welcher Insel welcher Teil-Workflow zu finden ist, und welche Abhängigkeiten zwischen Teil-Workflows bestehen.

5.1 Anpassung auf Inselseite

Zuerst widmen wir uns den notwendigen Anpassungen auf der Inselseite. Der Idealfall wäre natürlich, wenn wir die auf den Inseln vorliegenden Teil-Workflows unverändert einbinden könnten. Leider klappt das nicht ganz so einfach. Um die Kommunikation über den Koordinator abwickeln zu können, muß der

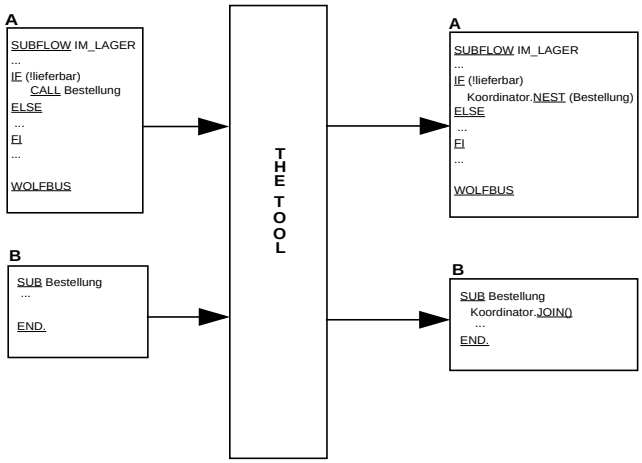
Aufruf des gewünschten Teil-Workflows angepaßt werden. Erfreulicherweise können wir diese Anpassung jedoch so gestalten, daß sie von jedem WfMS verkraftet werden kann. Insbesondere ist kein Eingriff in die Kernfunktionalität notwendig.

Abbildung 8 Verarbeitung



In Beispiel 2 ist die Vorgehensweise zu sehen. Der Workflow „IM_LAGER“ auf Insel A enthält eine Verzweigung zum Workflow „Bestellung“ auf Insel B. Das Analyse- und Transformationstool (im Moment noch als „the tool“ bezeichnet) identifiziert diese Stelle und ersetzt sie durch einen geeigneten Aufruf einer Funktion, die von der Koordinator-API angeboten wird. In diesem Fall wird „Bestellung“ als Auftrag („nested“) ausgeführt. Eine ausführliche Beschreibung der Koordinator-API folgt später.

Beispiel 2 Transformation der Insel-seitigen Sub-Workflows

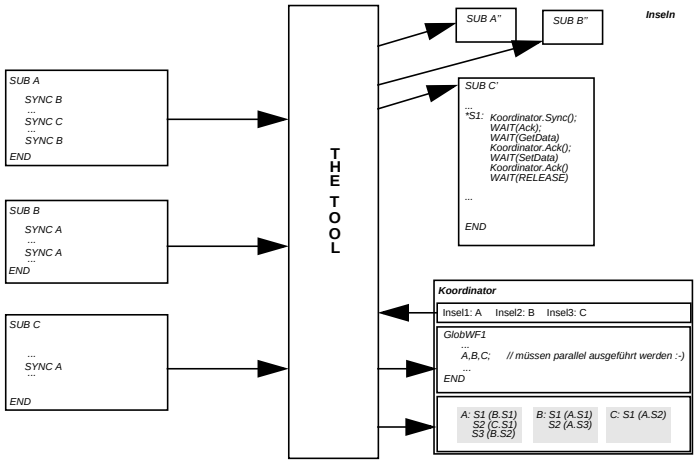


5.2 Gestaltung des Koordinators

Nach der Transformation der Teil-Workflows stehen uns die „Bauteile“ für den globalen Workflow zur Verfügung. Es fehlt jedoch noch die Beschreibung, die dem Koordinator die Ausführung ermöglicht. „The tool“ übernimmt auch diese Aufgabe. Aus den über die Sub-Workflows bekannten Informationen und der Integrationsvorschrift generiert es Kontroll- und Datenfluß- und Abhängigkeitsbeschreibungen. In Beispiel 3 werden drei Sub-Workflows miteinander synchronisiert. Die Sub-Workflows werden, wie oben beschrieben, von „the tool“ transformiert, die Synchronisationspunkte durch Aufrufe eines geeigneten Protokolls ersetzt. Da sich die drei Workflows nur synchronisieren können, wenn sie gleichzeitig aktiv sind, ist im Kontrollfluß ihre parallele Ausführung vorgesehen. Zusätzlich werden die Beziehungen zwischen den Inseln modelliert.

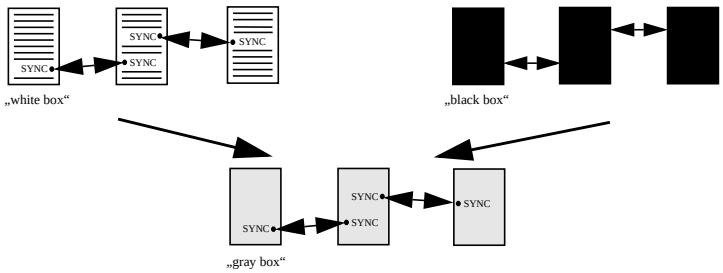
Diese Zusatzinformationen sind notwendig, um bei der späteren Ausführung des Workflows an diesen Stellen richtig reagieren zu können. Wird beispielsweise ein Synchronisationspunkt erreicht, so muß der Koordinator natürlich feststellen können, auf welchen zugehörigen Partner-Workflow er warten muß.

Beispiel 3 Erzeugen eines globalen Workflows



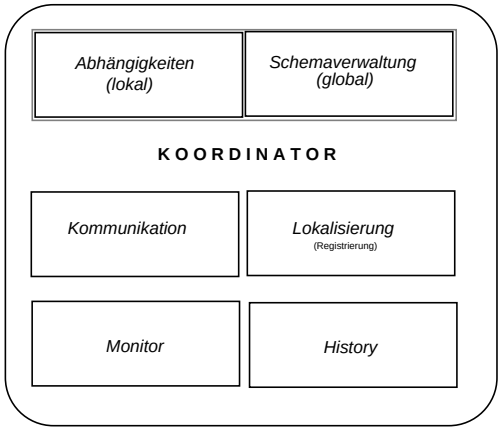
Da bei der Synchronisation innerhalb eines Workflows mehrere Synchronisationspunkte vorhanden sein können, müssen wir dies bei der Modellierung auch entsprechend berücksichtigen können. Den Sub-Workflow als „black box“ zu betrachten, reicht nicht, da wir so keine eindeutige Zuordnung der Synchronisationspunkte vornehmen können. Eine „white box“-Betrachtung ist nicht immer möglich und liefert auch viel zu viel nicht benötigte Informationen. Was zwischen den Synchronisationspunkten passiert, ist Sache der Insel und interessiert den Koordinator nicht. Daher wählen wir (wie in Abbildung 9 skizziert) einen Mittelweg: die „gray box“ beschreibt nur, welche Synchronisationspunkte ein Workflow anbietet (SUB A synchronisiert sich z. B. an drei Stellen, die entsprechend mit S1 bis S3 gekennzeichnet werden). Mit ihrer Hilfe lassen sich nun die zusammengehörigen Synchronisationspunkte zuordnen.

Abbildung 9 Idee der „gray box“



Aufgrund dieser Zuordnung läßt sich auch eine erste, sehr einfache Fehlerkontrolle bzgl. der Spezifikation des globalen Wf-Schemas durchführen. Wenn Synchronisationspunkte ohne Gegenstück existieren, ist die Spezifikation offensichtlich fehlerhaft. Zusätzlich sollten Synchronisationspunkte nur auf „top level“ zulässig sein, d. h., nicht durch Kontrollstrukturen wie „IF“ gekapselt werden dürfen. Nur so kann sichergestellt werden, daß keiner der Synchronisationspartner vergeblich wartet, weil der Partner (z. B. wegen einer fehlerhaften Bedingung) den passenden Synchronisationspunkt nicht erreichen kann. Mit dieser Vorgehensweise haben wir ein Instrument zur Deadlock-Erkennung/Vermeidung, mit dem wir nicht nur einfache Fehler erkennen können (fehlende Synchronisationspartner), sondern (ähnlich den Wartegraphen aus [RHGL97]) auch zyklische Wartesituationen aufspüren. Nach Aufstellen des Synchronisationsgraphen lassen sich Zyklen bestimmen. Die Knoten dieser Zyklen können sich gegebenfalls reihum blockieren. Da uns aber auch Informationen über den zeitlichen Ablauf der Synchronisation zur Verfügung steht („gray box“), läßt sich mit einem einfachen Algorithmus feststellen, ob tatsächlich ein Deadlock vorliegt!

Abbildung 10 Komponenten des Koordinators



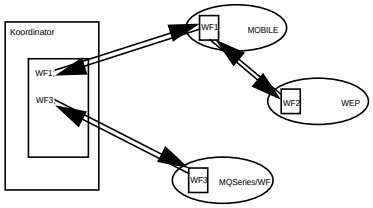
Das Wissen um das globale Schema (Kontroll- und Datenfluß) und die Abhängigkeiten zwischen den Sub-Workflows muß im Koordinator abgelegt werden; wie im vorangehenden Abschnitt beschrieben, liefert uns „the tool“ alle notwendigen Daten. Der Koordinator wird nun um zwei Komponenten erweitert, die die Verwaltung dieser Daten übernehmen: In Abbildung 10 sind sie mit *Abhängigkeiten* und *Schemaverwaltung* bezeichnet.
Im folgenden sollen nun noch weitere Ideen zur Rolle des Koordinators vorgestellt werden, die sich zur Zeit jedoch noch im Anfangsstadium befinden. Wir werden sie in zukünftigen Arbeiten weiterverfolgen.

5.3 Abstufung der Einbindung

Bei der Integration werden Sub-Workflows von verschiedenen Inseln zu einem globalen Workflow zusamengefügt. Sub-Workflows können untereinander Abhängigkeiten besitzen. In Abbildung 11 ist dieses Szenario nocheinmal dargestellt: Der Koordinator startet WF1, der wiederum einen Auftrag an WF2 vergibt. Und genau an dieser Stelle ergibt sich ein Problem: welche Rolle spielt der Koordinator bei diesem Starten des Auftrags? Im wesentlichen lassen sich 4 Fälle unterscheiden, die wir im folgenden näher betrachten wollen:

1. keine Koordinatorbeteiligung,
2. Meldung,
3. Kommunikation via Koordinator,
4. feineres Granulat.

Abbildung 11 Abhängigkeit zwischen Sub-Workflows



5.3.1 Keine Koordinatorbeteiligung

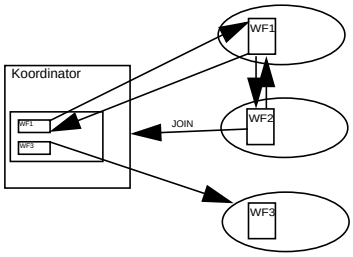
Voraussetzung für dieses Szenario ist, daß die beteiligten Inseln Interoperabilität zwischen Workflows direkt unterstützen, d. h., die Abhängigkeit zwischen den beiden Workflows kann direkt von den lokalen WfMS aufgelöst werden. Ein Eingreifen des Koordinators ist nicht notwendig. Der gerufene Sub-Workflow taucht daher in keiner Weise im globalen Schema auf.

5.3.2 Meldung

Es sind verschiedene Gründe denkbar, warum der Koordinator trotz der Fähigkeit eines Systems, Abhängigkeiten selbständig aufzulösen, beteiligt werden sollte. Monitoring könnte dies z. B. erfordern oder

das Mitprotokollieren in der History. Diese Beteiligung kann beispielsweise über eine Meldung des gerufenen Workflows an den Koordinator geschehen (Abbildung 12). Die API des Koordinators bietet mit Hilfe von „Join“ einem Workflow die Möglichkeit, seine Existenz und seine Ausführung mitzuteilen. Somit kann er auch im globalen Schema auftauchen und beispielsweise beim Monitoring oder bei der History berücksichtigt werden.

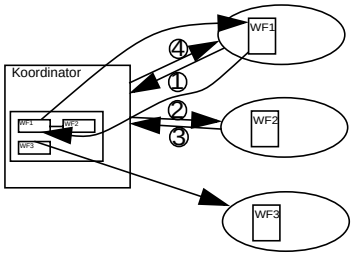
Abbildung 12 Meldemechanismus



5.3.3 Kommunikation über den Koordinator

Die wenigsten WfMS unterstützen Interoperabilität direkt. Folglich muß die Kommunikation über die vom Koordinator angebotene Funktionalität abgewickelt werden. Ein Auftrag wird in der in Abbildung 13 dargestellten Art und Weise abgewickelt. WF1 wird gestartet. Der Auftrag an WF2 wird initiiert, indem eine entsprechende API-Funktion des Koordinators aufgerufen wird (1). Dieser erzeugt und startet auf der zugehörigen Insel ein Exemplar von WF2 (2), empfängt das Ergebnis (3) und leitet dieses an den rufenden Workflow weiter (4). Dieser kann anschließend weiter ausgeführt werden.

Abbildung 13 Kommunikation via Koordinator



5.3.4 Verfeinerung der Granularität

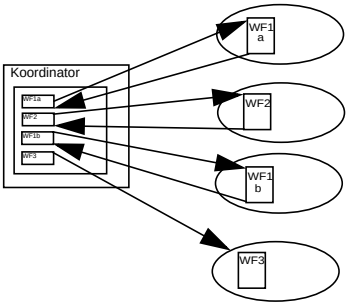
Eine weitere Möglichkeit, eine Abhängigkeit aufzulösen, besteht darin, den rufenden Workflow aufzuspalten, und den gerufenenen Workflow in den globalen Kontrollfluß/Datenfluß zu integrieren. Der Sub-Workflow wird somit vollständig in das globale Schema eingebunden und die Initiative zum Start der einzelnen Teil-Workflows geht nur vom Koordinator aus. Diese Vorgehensweise gibt dem Koordinator die maximale Kontrolle, erfordert aber auch die größte Veränderung an den auf den Inseln schon vorliegenden Schemata. Die bisher definierten Workflows müssen in weitere Teil-Workflows aufgeteilt werden. Problematisch wird dies, wenn eine starke Fragmentierung der Teil-Workflows stattfindet, so daß eine große Anzahl sehr kleiner Workflows entsteht.

5.3.5 Einsatz dieser Szenarien

Die ersten beiden betrachteten Fälle stellen bereits hohe Anforderungen an die beteiligten Inselsysteme. Diese müssen in der Lage sein, selbständig kommunizieren zu können. Bei allen anderen Systemen kommen Fall 3 oder 4 zum Tragen. Im Moment ist noch nicht klar abzuschätzen, welche Rolle diese vier

Fälle (insbesondere die beiden letzten) in der Praxis spielen. Dazu ist eine umfassende Analyse existierender Szenarien notwendig, die bisher noch nicht durchgeführt werden konnte. Dies soll aber geschehen, sobald geeignetes Material vorliegt.

Abbildung 14 Verfeinerung der Granularität

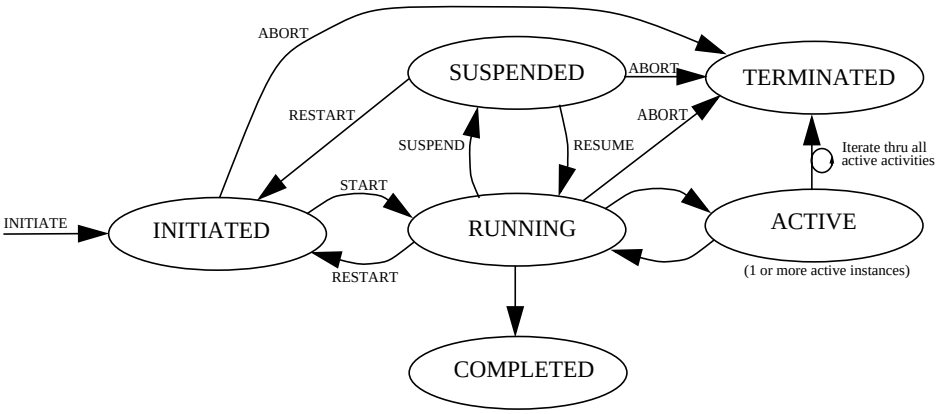


6. WF-Instanz-Zustand

Während der Ausführung einer Workflow-Instanz durch das WfMS kann ein Prozeß sich in verschiedenen Zuständen befinden. Ein Zustandsübergang kann durch ein Ereignis von außen ausgelöst werden (z. B. das Beenden einer Applikation) oder durch bestimmte Kontrollentscheidungen der Wf-Engine (beispielsweise Weiterschalten zur nächsten Aktivität in einem Prozeß). Als Basiszustände werden in [WfMC95] folgende 6 Zustände identifiziert:

- *initiated:* Eine Wf-Instanz wurde bereits erzeugt, aber nicht alle Startbedingungen sind erfüllt,
- *running:* Die Ausführung der Wf-Instanz wurde gestartet, jede ihrer Aktivitäten kann ausgeführt werden, jedoch wurde noch keine davon gestartet.
- *active:* Eine oder mehrere Aktivitäten der Wf-Instanz wurden gestartet.
- *suspended:* Die Workflow-Instanz ist untätig, keine weiteren Aktivitäten können gestartet werden, bevor nicht (mit Hilfe von „resume“) in den Zustand „running“ zurückgekehrt wurde.
- *completed:* Die Wf-Instanz erfüllt alle Bedingungen zur normalen Beendigung. Nach Abschluß der noch notwendigen Operationen (Logging, Statistiken schreiben) wird die Instanz gelöscht.
- *terminated:* die Ausführung wurde beendet, bevor das normale Ende erreicht werden konnte. Die notwendigen Operationen (Error-Log, Schreiben von Recovery-Daten) werden durchgeführt, danach wird die Instanz gelöscht.

Abbildung 15 Zustand-Übergangsgraph nach WfMC



Die möglichen Zustandsübergänge sind in Abbildung 15 grafisch dargestellt. Da Aktivitäten im allgemeinen nicht unterbrechbar sind, ist es nicht möglich SUSPEND und RESTART im Zustand ACTIVE aufzurufen. Dazu muß zuerst in den Zustand RUNNING zurückgekehrt werden.

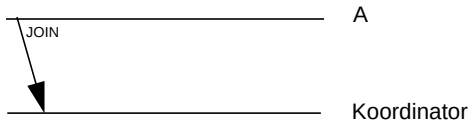
7. Protokolle

Für die Kommunikation zwischen Insel und Koordinator brauchen wir entsprechende Protokolle. Wir wollen nun die Abläufe zwischen Koordinator und Insel-Workflow betrachten. In dieser Diskussion wird noch keinerlei Fehlerbehandlung berücksichtigt, auch die zu übergebenden Parameter müssen noch festgelegt werden. Die hier gezeigten Protokolle decken sich weitgehend mit den in Abschnitt 3 beschriebenen Abhängigkeiten.

7.1 Meldung

Hierbei handelt es sich um das einfachste Szenario. Der neu gestartete Workflow auf Insel A meldet dem Koordinator seine Teilnahme. Da er unabhängig von einer Reaktion des Koordinators weiterlaufen kann, ist an dieser Stelle keine weitere Interaktion notwendig.

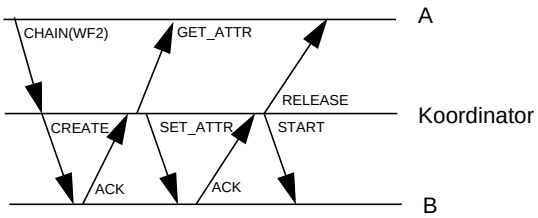
Abbildung 16 Meldung („Join“)



7.2 Anordnung

Bei der Anordnung soll auf Initiative eines bereits laufenden Workflows auf Insel A ein weiterer Workflow auf einer anderen Insel B gestartet werden. Nach dem Erzeugen und Starten dieses neuen Workflow-Exemplars setzt der rufende Workflow seinen Lauf fort, ohne sich um das weitere Schicksal des Sohnes auf B zu kümmern („Fire & Forget“).
Der rufende Workflow auf Insel A wendet sich an den Koordinator (mit Hilfe dem von diesem angebotenen API). Dieser lokalisiert die Zielinsel, erzeugt ein neues Exemplar des gewünschten Workflow-Typs, überträgt die zur Ausführung notwendigen Informationen und veranlaßt den Start. Der Workflow auf A wird gleichzeitig freigegeben und kann ohne weitere Verzögerung weiterlaufen. Das zugehörige Protokoll ist in Abbildung 17 zu sehen. Auf Insel A wird die von der Koordinator-API angebotene Methode „CHAIN“ benutzt, um dem Koordinator mit der Erzeugung einer neuen Anordnung zu beauftragen. Der rufende Workflow geht danach in einen Wartezustand. Die Realisierung dieses Wartens ist allerdings nicht ganz unproblematisch: Da die Kommunikation mit dem Koordinator über eine Applikation durchgeführt wird, befindet sich die WF-Instanz im Status „ACTIVE“. Es ist also leider nicht möglich, mit „SUSPEND“ und „RESUME“ zu arbeiten (siehe Abschnitt 6). Der blockierte Workflow wartet aktiv auf das Eintreffen der „RELEASE“-Nachricht. Wie dieses „BUSY-WAITING“-Problem entschärft werden kann, muß noch näher untersucht werden.

Abbildung 17 Anordnung („Chain“)



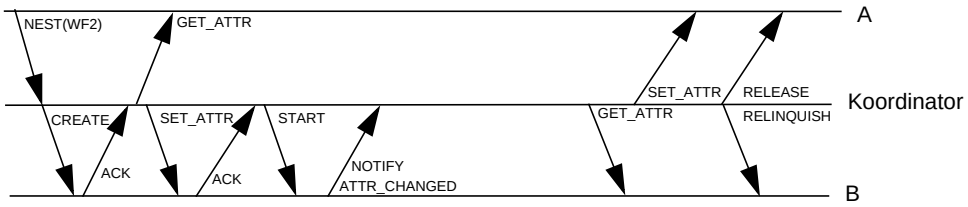
Der Koordinator erzeugt auf der Zielinsel B mit Hilfe der Insel-API eine Instanz des gewünschten Workflowtyps. Nach erfolgreicher Rückmeldung überträgt der Koordinator die zur Ausführung benötigten

Daten mit GET_ATTR und SET_ATTR. Anschließend wird der Workflow auf B gestartet. Mit Hilfe von RELEASE wird der Wartezustand des A-Workflows beendet. Um die Blockadezeit zu verkürzen, kann die RELEASE-Nachricht auch früher gesendet werden. Dies muß jedoch im Einklang mit einer evtl. noch einzuführenden Fehlerbehandlung geschehen.

7.3 Auftrag

Im Gegensatz zur Anordnung ist beim Auftrag das Ergebnis des gerufenen Workflows von Bedeutung für die weitere Ausführung des rufenden Ablaufs. Der Anfang ist analog zur Anordnung: Der Workflow auf Insel A wendet sich an den Koordinator. Dieser erzeugt ein neues Workflow-Exemplar auf der Zielinsel B, initiiert und startet es. Ist der Workflow auf B abgearbeitet, wird der Koordinator über die Gültigkeit der veränderten Daten in Kenntnis gesetzt. Der Koordinator holt sich die erforderlichen Informationen von B und reicht sie an den wartenden Workflow auf A weiter. Anschließend wird die Blockade des A-Workflows aufgehoben, d. h., dieser kann mit den gelieferten Ergebnissen weiter ausgeführt werden. Zum Protokoll: Gestartet wird der Auftrag mit einem Aufruf der Methode NEST des Koordinators. Nach dem Erzeugen einer neuen Instanz auf der Zielinsel B und dem Übertragen der notwendigen Daten von A nach B (GET_ATTR, SET_ATTR). Der Workflow auf B wird gestartet (START), der Workflow auf A bleibt in einem Wartezustand, bis ein Ergebnis eintrifft. Hat der Workflow B seine Arbeit geleistet, wird mit NOTIFY_ATTR_CHANGED der Koordinator davon in Kenntnis gesetzt. Das Ergebnis wird wieder mittels GET_ATTR, SET_ATTR zurückkopiert, anschließend wird der Workflow auf A aus seinem Wartezustand befreit (RELEASE). Der Workflow auf B bekommt eine RELINQUISH-Nachricht, die ihm signalisiert, daß diese Instanz nicht länger benötigt wird.

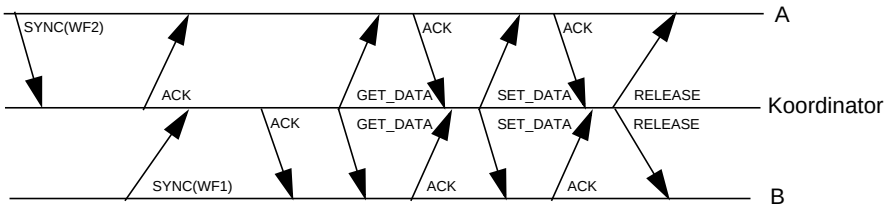
Abbildung 18 Auftrag („Nest“)



7.4 Synchronisation

Bei der Synchronisation tauschen zwei schon aktive Workflows an definierten Synchronisationspunkten miteinander Daten aus. Bei Erreichen eines dieser Punkte meldet der entsprechende Workflow sein Eintreffen dem Koordinator (SYNC) und wird blockiert. Sobald der Synchronisationspartner sich ebenfalls beim Koordinator gemeldet hat, holt dieser die auszutauschenden Daten von beiden Inseln ab (GET_ATTR) und leitet sie an den entsprechenden Partner (SET_ATTR) weiter. Sobald diese den Empfang quittiert haben, kann die Blockade aufgehoben werden (RELEASE) und beide Workflows setzen ihre Verarbeitung fort.

Abbildung 19 Synchronisation



8. Fehlerfälle

Bei den besprochenen Protokollen haben wir eine sichere Kommunikation zu Grunde gelegt, d. h., keine Nachrichten gehen verloren, die zeitliche Reihenfolge bleibt erhalten. Trotz dieser Annahme sind verschiedene Fehlersituationen denkbar, die hier beschrieben werden sollen.

8.1 Protokollfehler

Bei einem Protokollfehler wird die falsche Nachricht geschickt. Denkbar sind zwei Fälle: Die Nachricht entspricht nicht den in der API vorgesehenen Nachrichten („Syntax-Fehler“), oder die Nachricht kommt zum falschen Zeitpunkt an (beispielsweise ein SET_ATTR vor einem CREATE).
Da die Umsetzung der Protokolle auf Inselseite mit Hilfe von „the Tool“ generiert werden, sollten diese Fehler nicht auftreten (vorausgesetzt, „the Tool“ funktioniert zuverlässig). Treten sie dennoch auf, so ist ein „manuelles“ Eingreifen wohl unumgänglich.

8.2 Synchronisationsfehler

Ein Synchronisationsfehler liegt vor, wenn zyklische Warteabhängigkeiten vorliegen. Mit Hilfe der Abhängigkeitsverwaltung des Koordinators („gray boxes“) lassen sich diese mit geeigneten Algorithmen frühzeitig erkennen. Sollte es zu einer Verklemmung kommen, so muß diese erst als solche erkannt werden. Der pragmatischste Ansatz ist wohl, nach einer gewissen Wartezeit eine Warnung auszuwerfen, daß eine Verklemmung vorliegen könnte. Auch in diesem Fall sind Überprüfung und ggf. Eingreifen durch den Menschen notwendig.

8.3 Laufzeitfehler

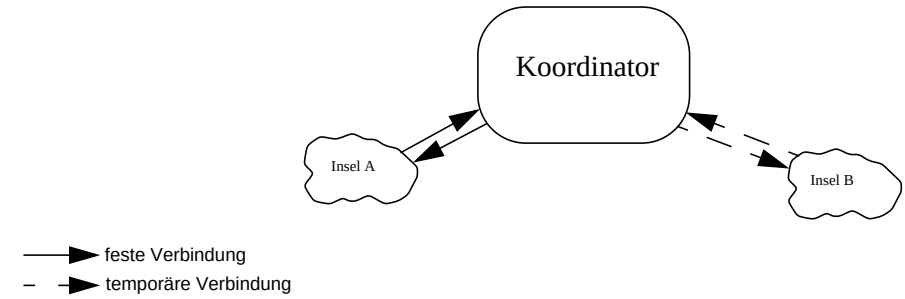
Ein Workflow auf einer Zielinsel kann nicht ausgeführt werden (Lokalisierungsfehler) oder endet fehlerhaft. Beispielsweise kann die Engine auf der Zielinsel ausgefallen sein. Auch hier empfiehlt sich ein manueller Eingriff.

Es sind noch weitere Fehler denkbar, die beispielsweise technischer Natur sind (Ausfall des Koordinators), auf die in diesem Papier jedoch noch nicht eingegangen werden soll. Bei den oben beschriebenen Fehlerklassen scheint ein manuelles Eingreifen unumgänglich zu sein, eine automatisierte Fehlerbehandlung erscheint wenig sinnvoll.

9. Mobile Inseln

Bisher sind wir immer davon ausgegangen, daß alle Inseln jederzeit dem System zur Verfügung stehen. Interessant ist aber auch die Frage, inwieweit „mobile Inseln“ integriert werden können. Tragbare Geräte sind inzwischen sehr leistungsfähig und verbreiten sich immer mehr. Gerade im Außendienst sind sie nicht mehr wegzudenken. Wollen wir mobile Systeme integrieren, tauchen jedoch eine Reihe neuer Probleme auf.

Abbildung 20 mobile Insel



Mobile Inseln sind nicht immer erreichbar. Wir gehen vielmehr von dem folgenden Szenario aus: Die Insel nimmt Kontakt mit dem Koordinator auf, liefert diesem ggf. eine Rückmeldung über bereits ausgeführte Aktionen, bekommt neue Anweisungen übermittelt und beendet die Verbindung wieder. Der Erreichbarkeitszeitraum ist somit eng begrenzt. Diese Vorgehensweise erfordert offensichtlich asynchrone Kommunikation zwischen Koordinator und Insel. Der nächste Kontakt zur Insel ist schwer vor-

aussagbar. Während dies bei Anordnungen auch unkritisch ist, so ergibt sich bei Aufträgen/ Synchronisation ein Problem mit dem blockierten rufenden Workflow.
Hier ist zu prüfen, ob diese Abhängigkeiten für Workflows auf mobilen Einheiten überhaupt Sinn machen, oder ob sie schlicht und ergreifend verboten werden sollen.
Eine weitere Idee besteht darin, Workflow-Typen auf mehreren (mobilen) Inseln anzubieten. Der Koordinator kann dann je nach Verfügbarkeit einer Insel den Subworkflow entsprechend delegieren.

10. Zusammenfassung und Ausblick

Aufgrund der vielen angebotenen und eingesetzten WfMS ist es nicht ganz einfach, Workflows für firmenübergreifende Prozesse zu definieren. In diesem Papier haben wir erste Ideen präsentiert, wie dieses Problem angegangen werden kann. Wir haben den Begriff der „Workflow-Insel“ eingeführt und verstehen darunter ein beliebiges WfMS mit zugehörigen Wf-Schemata, die zur Definition eines globalen Schemas herangezogen werden können. Weiterhin haben wir die „Koordinator-Ebene“ als übergeordnete Stufe, auf der das globale Schema definiert werden kann, kennengelernt.
Zur Verteilung der Sub-Wf-Schemata auf die Inseln wurden zwei Verfahren vorgestellt. Bei der Integration werden schon vorhandene Schemata (leicht modifiziert) eingebunden. Bei der Delegation hingegen wird der Gesamtablauf zentral definiert, anschließend werden daraus Teil-Schemata abgeleitet und auf den Inseln (in einer dem lokalen WfMS verständlichen Form) abgelegt. Im allgemeinen ist mit einer Mischform aus beiden Verfahren zu rechnen.
Wir haben die verschiedenen Formen der Interoperation vorgestellt, wie sie in [WfMC96] beschrieben werden. Weiterhin haben wir motiviert, daß neben Anordnung, Auftrag und Synchronisation noch weitere Szenarien denkbar sind. Ein Beispiel hierfür ist der „Wettbewerb“, bei dem die Anzahl der teilnehmenden Inseln erst zur Laufzeit feststeht.
Nach kurzer Betrachtung der Aufgaben, die eine Koordinationskomponente erfüllen muß (globales Schema interpretieren, initiieren lokaler Wf-Instanzen, Abhängigkeitskontrolle, Kommunikation zwischen lokalen Wf-Instanzen, globalen Zustand feststellen, Monitoring, History, Administration), wurden kurz eine zentrale und eine dezentrale Lösung diskutiert. Wir haben uns danach für den zentralen Ansatz entschieden.
Weiterhin wurde gezeigt, welche Schritte bei der Integration durchzuführen sind, um zu einem globalen Schema zu gelangen. Dazu wurde das Werkzeug „the tool“ vorgestellt und verschiedene Komponenten des Koordinators identifiziert.
Ein weiteres angedachtes Problem ergibt sich bei der direkten Kommunikation zwischen Sub-Workflows. Hier haben wir verschiedene erste Ideen präsentiert, inwieweit der Koordinator eingebunden werden kann, bzw. muß.
Wir haben das Zustandsmodell der WfMC betrachtet und anschließend Protokolle zur Kommunikation zwischen Insel und Koordinator eingeführt. Dabei wurde das Problem erkannt, daß ein blockierter Workflow leider nicht in den Zustand „SUSPENDED“ überführt werden kann.
Wir haben weiterhin verschieden Fehlerfälle identifiziert. Fast immer ist beim Auftreten von Fehlern ein manueller Eingriff notwendig, automatisierte Maßnahmen scheinen wenig sinnvoll.
Zum Abschluß haben wir noch kurz angedacht, inwiefern „mobile Inseln“ eingebunden werden können.
Der nächste Schritt soll nun sein, unsere hier dargestellte Ideen gegen Anforderungen aus der Industrie zu prüfen, um die Praxistauglichkeit zu testen. Das Kommunikationsmodell muß verfeinert werden, die Probleme und Möglichkeiten mobiler Inseln bieten ebenfalls noch ein großes Betätigungsfeld.

11. Literatur

[BDS98] Thomas Beuter, Peter Dadam, Peter Schneider, *The WEP Model: Adequate Workflow-Management for Engineering Processes*, ECEC1998

[HLKP2000] Hong, Lee, Kim, Paik, *A Web-Based Transactional Workflow Monitoring System*, Juni 2000, Proc. 1st Int. Conf. on Web Information Systems Engineering (WISE 2000), Hongkong, pp. 149-156

[RHGL97] Rezende, Härder, Gloeckner, Lutze, *Detection Arcs for Deadlock Management in Nested Transactions and their Performance, in: Advances in Databases*, Proc. 15th British National Conference on Databases (BNCOD'97), London, U.K., LNCS 1271, Springer, July 1997, pp. 54-68.

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

[WfMC95] Workflow Management Coalition, The *Workflow Reference Model*, Jan. 1995, Document Number WfMC TC00-1003, www.aiim.org/wfmc/mainframe.htm

[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