# Entwurf einer transaktionalen Workflow-Architektur zur sicheren Nachweisführung

Stefan Mark[1] · Stephan Groß[1] · Jost Kannegieser[2] · Jörg Kebbedies[2]

[1]Technische Universität Dresden
Institut für Systemarchitektur
01062 Dresden
stefan.mark@inf.tu-dresden.de, stephan.gross@tu-dresden.de

[2]secunet Security Networks AG
01067 Dresden
{jost.kannegieser, joerg.kebbedies}@secunet.com

## Zusammenfassung

Workflow-Management-Systeme (WfMS) unterstützen Firmen und Behörden bei der Abwicklung ihrer elektronischen Geschäftsprozesse. In diesen Prozessen werden u.a. Verschlusssachen und andere vertrauliche Dokumente verarbeitet. Neben der vertraulichen Übertragung besteht dabei die Notwendigkeit, den Zugriff auf diese Dokumente verbindlich einem Nutzer zuzuordnen und zu protokollieren. Aktuelle WfMS gewährleisten dies nur bedingt.

Die vorliegende Arbeit gibt einen Überblick über existierende Systeme und erklärt, warum diese die Anforderungen nicht erfüllen. Danach werden die Anforderungen an ein WfMS zur transaktionalen Nachweisführung präzisiert. Im Anschluß daran wird ein transaktionales WfMS, aufbauend auf dem Workflow-Referenzmodell der Workflow Management Coalition und dem OTS Transaktionsstandard, entworfen, das diesen Anforderungen gerecht wird.

## 1 Einleitung

Unternehmen und Behörden sind heute zwingend auf IT-Systeme für die Abwicklung ihrer Geschäftsprozesse angewiesen. Deren operativ-technische Realisierung wird in sogenannten *Arbeitsabläufen* (engl. *Workflows*) definiert, die eine festgelegte Abfolge von Aktivitäten innerhalb einer Organisation beschreiben. Nach [WMC99] ist ein Workflow „ein teilweise oder vollständig automatisierter Geschäftsprozess, in dem Dokumente, Informationen oder Aufgaben zwischen Teilnehmern entsprechend einer Menge von Ausführungsregeln übertragen werden“. Er legt also die für eine konkrete Zielsetzung notwendige Verarbeitung von Daten mittels verschiedener Anwendungen fest. Die hierfür notwendigen Datenaustauschprozesse sind in der Regel auf unterschiedliche Systemkomponenten verteilt, woraus sich Anforderungen an die Vertraulichkeit und Integrität der dafür notwendigen Datenübertragung ableiten lassen. Bisher wurden hierbei meist nur die Kommunikationsinhalte betrachtet. Neue rechtliche Bestimmungen und die wachsende Komplexität der eingesetzten Systeme erfordern heute je-

P. Horster (Hrsg.) · D•A•CH Security 2008 · syssec (2008) 1-10.

doch eine Ausdehnung dieser Schutzanforderungen auf die Kommunikationsteilnehmer, um etwa eine sichere Zurechenbarkeit von sicherheitsrelevanten Aktionen und handelnden Akteuren zu gewährleisten. Beispiele hierfür finden sich bei der Verarbeitung von Verschlusssachen im behördlichen Umfeld, aber auch bei der Kontrolle von Informationsflüssen in Unternehmen zum Schutz von Betriebsgeheimnissen.

Das Beispiel eines Workflows, der die Verarbeitung von vertraulichen Dokumenten protokolliert, ist in Abbildung 1 dargestellt. Im ersten Schritt muss sich der Nutzer authentisieren, um seine Autorisation für die nachfolgenden Prozessschritte zu gewährleisten und außerdem Zurechenbarkeit bei deren Durchführung zu ermöglichen. Danach wird ein Dokument zuerst angesehen, anschließend bearbeitet und zuletzt versendet. Alle diese Aktionen sind nachweisrelevant und werden deshalb mit ihrer Protokollierungsoperation zu Transaktionen zusammengesetzt.

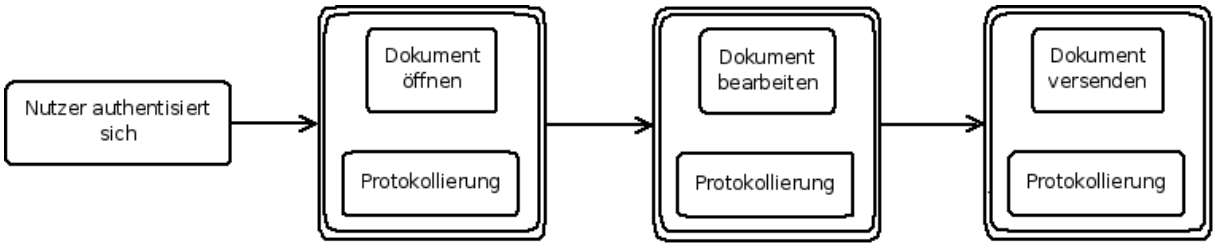


**Abbildung 1:** Beispielszenario eines transaktionalen Workflows

Herkömmliche WfMS sind für solche Szenarien nur bedingt tauglich, da sie zwar die Ausführung von Arbeitsabläufen organisieren und koordinieren, dabei jedoch die Behandlung von Fehler- und Ausnahmefällen nur unzureichend berücksichtigen. Aspekte der konsistenten und zuverlässigen Datenverwaltung werden dabei oft vernachlässigt. Ein Schlüsselkonzept zur Behebung dieses Mankos sind die ursprünglich aus dem Datenbankbereich bekannten Transaktionen, die sich an die Belange verteilter Umgebungen anpassen lassen.

*Transaktionen* kapseln Operationen zur Bearbeitung von Daten und realisieren die sogenannten ACID-Eigenschaften [GR92]: Die *Atomarität* (engl. *Atomicity*) gewährleistet, dass alle innerhalb einer Transaktion ausgeführten Operationen nach außen wie eine unteilbare Operation wirken. Falls eine der Teiloperationen fehlschlägt, schlägt auch die Transaktion fehl und alle Zwischenergebnisse werden verworfen. Die Eigenschaft der *Konsistenz* (engl. *Consistency*) sichert zu, dass sich alle Daten nach dem Abschluss einer Transaktion in einem widerspruchsfreien (konsistenten) Zustand befinden. Dazu werden Integritätsbedingungen definiert, die festlegen, wann ein Zustand konsistent ist. Die Eigenschaft der *Isolation* fordert, dass sich zwei oder mehr Transaktionen nicht gegenseitig beeinflussen. Die Eigenschaft der *Dauerhaftigkeit* (engl. *Durability*) legt schließlich fest, dass alle in einer Transaktion durchgeführten Änderungen am Ende der Transaktion persistent gespeichert werden. Durch diese Eigenschaft wird verhindert, dass gerade durchgeführte Änderungen verloren gehen.

Die vorliegende Arbeit baut auf einer Diplomarbeit zur transaktionsorientierten Steuerung in Workflows [Mar08] auf, welche bei der secunet Security Networks AG in Kooperation mit der Technischen Universität Dresden entstanden ist. Die Arbeit präsentiert den aktuellen Stand unserer laufenden Forschung zum Entwurf einer transaktionsorientierten Workflow-Architektur als Grundlage für ein Audit-System zur Überwachung von Workflow-Prozessen. Die vorgeschlagene Lösung gewährleistet die verbindliche Zurechenbarkeit von Operationen und handelnden Akteuren.

Diese Arbeit hat folgenden Aufbau: In Kapitel 2 werden zunächst existierende Arbeiten auf dem Gebiet der Transaktionen, Workflows und transaktionalen Workflows vorgestellt. In Kapitel 3 gehen wir dann auf die grundlegenden Voraussetzungen für ein transaktionsorientiertes Workflow-Management-System ein. Danach werden in Kapitel 4 relevante Grundkonzepte verteilter Transaktionen zusammengefasst und wir geben einen kurzen Einblick in das Thema Workflows. Der anschließend vorgestellte Architekturentwurf in Kapitel 5 basiert auf dem Referenzmodell der Workflow Management Coalition (WfMC) [Hol95] und zeigt den Aufbau eines WfMS, welches transaktionale Nachweisführung gewährleisten kann. In Kapitel 6 stellen wir vor, wie das beschriebene System in die Praxis umgesetzt werden kann und präsentieren erste Erfahrungen mit dem von uns entwickelten Prototypen.

# 2 Relevante Arbeiten und Motivation

Ein Konsortium, das sich mit der Standardisierung von Workflow-Management-Systemen befasst, ist die Workflow Management Coalition (WfMC). Unter dem Dach der WfMC wurde ein Workflow-Referenz-Modell [Hol95] erarbeitet, welches der Standard für aktuelle WfMS ist. Das Referenz-Modell wird durch die Spezifikationen der verschiedenen Schnittstellen eines WfMS ergänzt [WMC].

Damit ein WfMS einen Workflow ausführen kann, muss dieser in geeigneter Form beschrieben werden können. Von der WfMC wurde dazu die Workflow-Ausführungssprache XPDL [XPD05] entwickelt. Eine weitere weit verbreitete Workflow-Ausführungssprache ist BPEL [BPE07], welche Workflows auf der Basis von Webservices beschreiben kann. Beide Sprachen können aus einer BPMN Notation [BPM06] erzeugt werden, welche zur grafischen Beschreibung eines Workflows dient und damit die Entwicklung eines Workflows für den Nutzer vereinfacht.

Mit der Entwicklung der ersten Workflow-Management-Systeme entstand auch bald die Notwendigkeit von Transaktionen in Workflows. Dafür wurde die aus der Datenbankwelt bekannte Transaktion zur Unterstützung in WfMS integriert. Diese Workflowtransaktionen können durch die heute gängigen Workflowausführungssprachen (XPDL, BPEL, etc.) abgebildet werden.

Die für transaktionale WfMS verwendeten Transaktionsmodelle sind u.a. Sagas [GMS87], sowie flache und verschachtelte Transaktionen [Mos81]. Eine der ersten Arbeiten zu Transaktionen in WfMS findet sich in [RS95]. Ein Beispiel eines transaktionalen WfMS ist der von IBM entwickelte WebSphere Process Server [IBM].

Aktuelle Workflow-Management-Systeme gehen von der durchaus richtigen Annahme aus, dass Transaktionen in Workflows von langer Dauer sein können und nutzen deshalb meist Kompensation zur Einhaltung der ACID-Eigenschaften. Für die Protokollierung von sicherheitskritischen Vorgängen ist Kompensation jedoch nicht möglich, da nicht immer eine Kompensationsoperation existiert. Sind etwa vertrauliche Informationen erst einmal publik gemacht, lässt sich das i.d.R. nicht mehr zurück nehmen. Aus der Notwendigkeit für transaktionale Nachweisführung in Workflows und den fehlenden Mechanismen zu deren Umsetzung entstand die Idee zu dieser Arbeit.

# 3 Voraussetzungen

Bevor wir in die Grundlagen von Transaktionen und Workflows einsteigen, sollen zuerst die bereits genannten Anforderungen zur sicheren Nachweisführung in Workflows präzisiert werden.

Da die betrachteten nachweisrelevanten Operationen in einem Geschäftsprozess stattfinden, wird zuerst eine *Workflow-Architektur* als Grundvorraussetzung benötigt. Dazu gehört ein Workflow-Management-Systems, welches Transaktionen unterstützt. Diese Transaktionen setzen sich dann aus der nachzuweisenden Operation und der dazugehörigen Protokollierung zusammen.

Zudem muss es möglich sein, den Workflow zu beschreiben. Dazu wird eine *Workflow-Beschreibung* benötigt. Dafür soll vorzugsweise ein existierender Standard verwendet und gegebenenfalls erweitert werden, um die Interoperabilität mit existierenden WfMS zu gewährleisten. Außerdem muss die Workflow-Beschreibung die Transaktionssemantik kennen, damit die nachzuweisende Operation und das dazugehörige Audit transaktional beschrieben werden können.

Zur Unterstützung von Transaktionen im WfMS wird eine *Transaktionssteuerung* benötigt. Diese muss *verteilte Transaktionen* unterstützen, da auch Workflow-Aktivitäten über viele Systeme verteilt stattfinden können.

Zuletzt müssen noch *Schnittstellen für die am Workflow beteiligten Resourcen bzw. Komponenten* geschaffen werden, welche diesen die Teilnahme an einer Transaktion erlauben. Durch die Implementierung dieser Schnittstellen muss es dann möglich sein, eigene Aktivitäten (z.B. nachweisrelevante Operationen oder die Protokollierung) in einen transaktionalen Workflow einzubringen.

# 4 Grundlagen

## 4.1 Grundkonzepte verteilter Transaktionen

Verteilte Transaktionen erstrecken sich grundsätzlich über mehrere miteinander vernetzte Systeme. Sowohl das zugrunde liegende Netzwerk als auch die beteiligten Rechner können unabhängig voneinander Fehler aufweisen. Zur Umsetzung der ACID-Eigenschaften ist daher eine verteilte Koordination zwischen den beteiligten Instanzen erforderlich.

### 4.1.1 Commit- und Kompensationsprotokolle

Zur Erfüllung der Atomaritäts-Eigenschaft werden bei verteilten Transaktionen Kompensations- oder Commit-Protokolle verwendet.

Beim Commit-Protokoll werden alle beteiligten Ressourcen für die Dauer der Transaktion für den Zugriff durch andere Prozesse gesperrt. Änderungen werden nicht sofort angewendet, sondern von der entsprechenden Ressource erst einmal nur vorgemerkt. Diese Änderungen werden am Ende der Transaktion durchgeführt. Dazu existieren verschiedene Commit-Protokolle (z.B. 2 Phase Commit), welche dafür sorgen, dass die Änderungen nach außen atomar erscheinen.

Bei der Kompensation werden alle Änderungen sofort für die beteiligten Ressourcen angewendet. Falls in der Transaktion ein Fehler auftritt, existieren eine oder mehrere Kompensationsoperationen, welche die durchgeführten Änderungen semantisch rückgängig machen. Wenn alle beteiligten Ressourcen für die gesamte Dauer der Transaktion für Zugriffe durch andere Transaktionen gesperrt werden, erfüllt Kompensation die strikten ACID-Eigenschaften. Meist wird Kompensation jedoch ohne Sperren angewendet, da es auf diese Art seine Vorteile (Transaktionen blockieren sich nicht gegenseitig) gegenüber den Commit-Protokollen ausspielen kann. Es eignet sich damit für langandauernde Transaktionen. Im Fehlerfall kann es allerdings zu kaska-

dierten Transaktionsabbrüchen kommen. Kompensation kann zudem nur angewendet werden, wenn eine Kompensationsoperation existiert. Dies ist nicht immer der Fall.

Zur Lösung des Problems der transaktionalen Nachweisführung in Workflows haben wir uns für die Verwendung von Commit-Protokollen entschieden, da diese die Alles-oder-Nichts-Eigenschaft garantieren und weil für die meisten sicherheitskritischen Operationen keine Umkehroperationen existieren.

### 4.1.2 Transaktionsmodelle

In der Literatur finden sich verschiedene Transaktionsmodelle, welche je nach Einsatzgebiet gewisse Vor- oder Nachteile besitzen. Im Folgenden sollen zwei Modelle für verteilte Transaktionen vorgestellt werden.

Das in der Praxis am meisten verwendete Transaktionsmodell ist das *Flat Transaction* Modell. Es verwendet das 2 Phase Commit-Protokoll und wendet alle Änderungen erst am Ende der Transaktion an. Aufgrund seiner Einfachheit ist es zudem leicht zu implementieren.

Ein weiteres Modell sind *geschachtelte Transaktionen* (engl. *Nested Transactions*) [Mos81], die auf *Flat Transactions* aufbauen. Sie besitzen die Möglichkeit Subtransaktionen zu definieren, welche von der Gesamttransaktion unabhängig ein Commit ausführen oder zurücksetzen können. Im Gegensatz zu einer *Flat Transaction* muss im Fehlerfall nicht die ganze Transaktion, sondern nur die fehlgeschlagene Subtransaktion, rückgängig gemacht werden. Für die fehlgeschlagene Subtransaktion kann dann eine Alternativoperation ausgeführt werden.

## 4.2 Workflow-Management

Zur Definition, Erzeugung und Steuerung von Workflows werden Workflow-Management-Systeme eingesetzt. Eine gute Beschreibung eines WfMS liefert das Workflow Referenzmodell (Abbildung 2) der Workflow Management Coalition (WfMC).

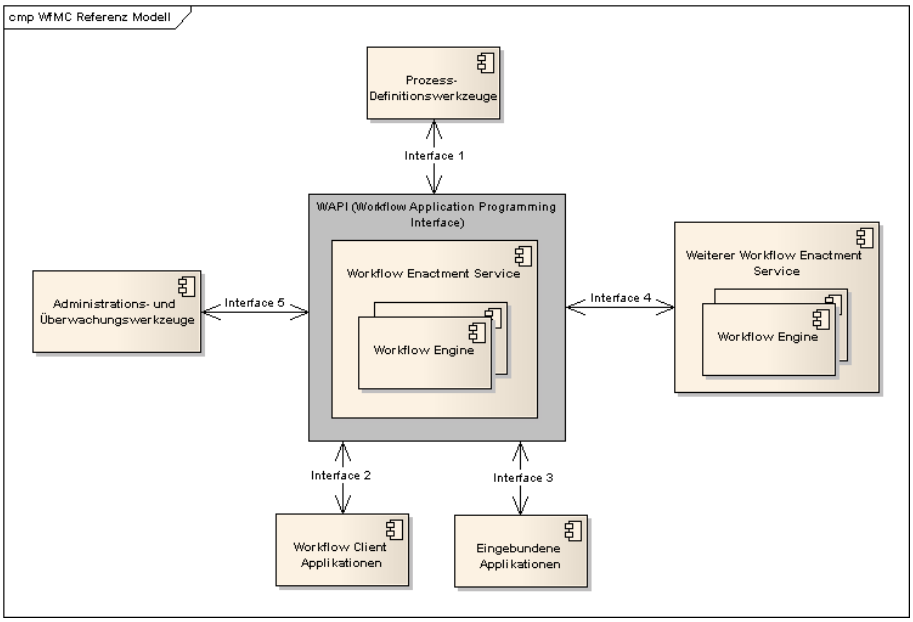


**Abbildung 2:** WfMC Referenzmodell [Hol95]

Dieses Modell besteht aus den Kernkomponenten *Workflow Enactment Service*, welcher die Laufzeitumgebung für Workflow-Engines bereitstellt, und mehreren *Workflow-Engines*, welche auf Funktionen des Workflow Enactment Service zurückgreifen und je eine Workflow-Definition interpretieren.

Darüber hinaus existieren fünf Schnittstellen, welche es erlauben, Administrations- und Überwachungswerkzeuge, Prozessdefinitionswerkzeuge, Workflow Client Applikationen zur Interaktion mit Nutzern, automatisierte Dienste, sowie andere Workflow Enactment Services mit in die Workflow-Abarbeitung einzubeziehen.

Um einen Workflow ausführen zu können, muss er zuerst beschrieben werden. Dazu wurde von der WfMC die XML Process Definition Language (XPDL) entwickelt, welche im Moment in ihrer zweiten Version vorliegt [XPD05]. Sie kann die Komponenten und Schnittstellen des WfMC Referenzmodells beschreiben und somit recht einfach in ein, auf diesem Modell basierendes, System integriert werden.

### 4.2.1 Unterstützung von Transaktionen

In der WfMC wurde die Notwendigkeit von Transaktionen in Workflows bereits erkannt. In der XPDL existiert die Möglichkeit, einen Subprozess als Transaktion zu definieren und eine Kompensationsoperation für den Fehlerfall anzugeben. Darüber hinaus werden jedoch keine konkreteren Angaben zur Transaktionsausführung gemacht. Die Interpretation des transaktionalen Subprozesses ist daher von der Implementierung des WfMS abhängig. Sowohl Kompensations-, als auch Commit-Protokolle sind dadurch realisierbar.

Neben XPDL existieren auch andere Standards zur Beschreibung von Geschäftsprozessen. Eine andere, weit verbreitete und auf der Komposition von Web Services beruhende Beschreibungssprache ist die Web Service Business Process Execution Language (WS-BPEL) [BPE07]. Auch diese Sprache unterstützt Transaktionen. Allerdings besteht nur die Möglichkeit der Kompensation, falls innerhalb eines vorher definierten Blocks eine Operation fehlschlägt. Transaktionen, welche auf dem Commit-Protokoll basieren, können dort nicht direkt abgebildet werden.

### 4.2.2 Unterstützung von Auditfunktionalität

Um sicherheitskritische Vorgänge innerhalb eines Workflows nachweisen zu können, benötigt ein WfMS Auditfunktionalität. Das WfMC Referenzmodell beschreibt bereits eine Auditschnittstelle, die allerdings nur Daten protokolliert, welche beim Aufruf der Schnittstellenfunktionen generiert werden. Die Protokollierung von Rollen-, Nutzer- oder Sicherheitsinformationen wird nicht beschrieben.

Um die verbindliche Protokollierung zu gewährleisten, muss das Audit zudem transaktionale Eigenschaften besitzen. Auch dies ist im WfMC Referenzmodell nicht vorgesehen und muss in unserer Architektur realisiert werden.

# 5 Architekturentwurf

Als Grundlage für den Entwurf eines transaktionalen WfMS dient das in Kapitel 4.2 beschriebene Referenzmodell der WfMC. Dieses Modell wird im folgenden Abschnitt näher beschrieben, bevor es anschließend um Transaktionskonzepte erweitert wird.

## 5.1 Aufbau eines Workflow-Management-Systems

Aufbauend auf dem WfMC Referenzmodell existiert ein Entwurf der Object Management Group (OMG), welcher dieses Workflowmodell auf der Grundlage von CORBA konkret spezifiziert [OMG00]. Die wichtigsten Schnittstellen dieses Modells sind in Abbildung 3 dargestellt.

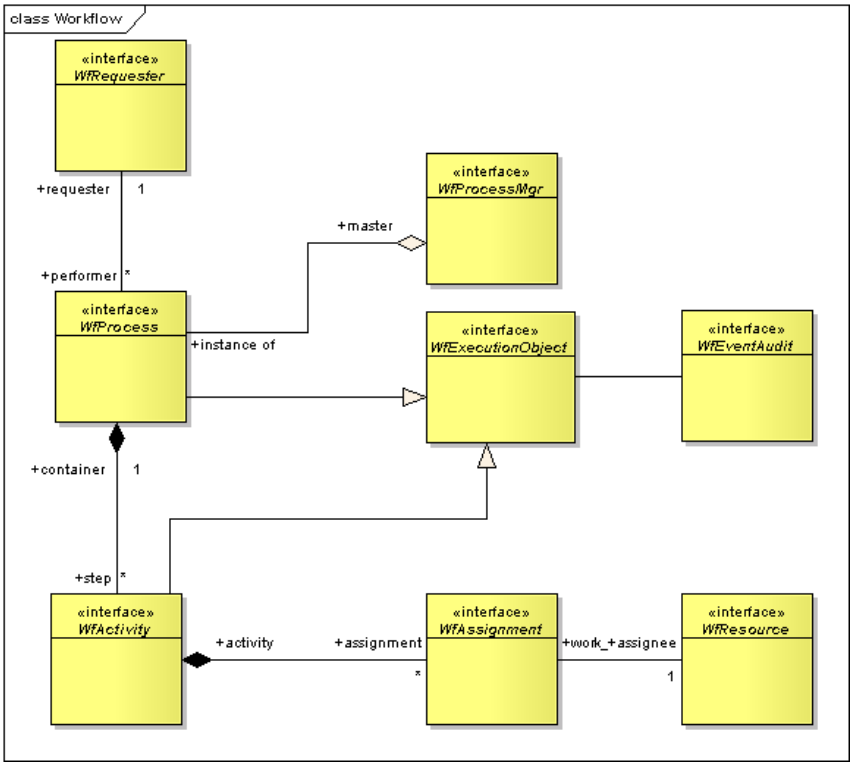


**Abbildung 3:** OMG Workflowmodell

Abbildung 4 verdeutlicht das Zusammenspiel der Komponenten des OMG Workflowmodells. Es zeigt, welche Schnittstellen zu welcher Komponente gehören und wie die einzelnen Schnittstellen miteinander verbunden sind. Der Ablauf eines aus zwei Schritten bestehenden Workflows wird durch die gepunkteten Pfeile dargestellt.

Die dargestellten Schnittstellen haben folgende Aufgaben:

**WfProcess:** Ein WfProcess stellt eine Instanz eines Workflowprozesses dar. Der WfProcess steuert den Workflow und legt fest, wann die einzelnen Aktivitäten ausgeführt werden. Um einen WfProcess zu instanziieren, kann z.B. eine Workflowbeschreibung verwendet werden.

**WfActivity:** Eine WfActivity ist die interne Repräsentation einer Workflowaktivität innerhalb der Workflow-Engine. Eine WfActivity steuert eine eingebundene Applikation. Diese Schnittstelle wird durch den WfProcess dazu veranlasst, eine eingebundene Applikation zu starten, zu unterbrechen, fortzusetzen oder abzubrechen.

**WfAssignment:** Ein WfAssignment stellt die Verbindung zwischen der WfActivity und der eingebundenen Applikation dar. Dazu besitzt das WfAssignment Möglichkeiten, eine Workflowaktivität aufzufinden.

**WfResource:** WfRessource stellt die Schnittstelle dar, welche den einheitlichen Zugriff auf alle eingebundenen Workflowaktivitäten ermöglicht.

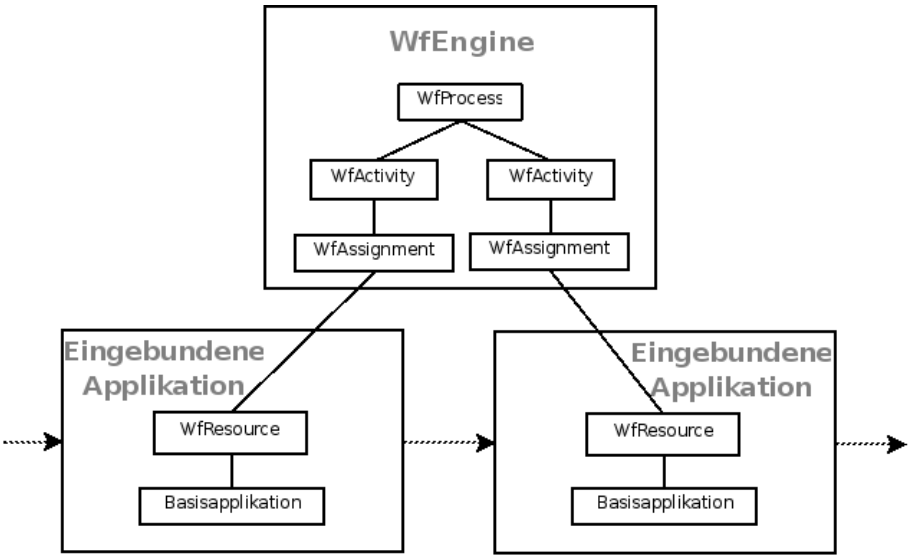


**Abbildung 4:** Workflow unter Verwendung des OMG Referenzmodells

**Basisapplikation:** Die Basisapplikation realisiert eine Workflowaktivität. Mit Hilfe der WfResource-Schnittstelle wird der Workflow-Engine ermöglicht, auf diese Applikation zuzugreifen.

## 5.2 Erweiterung eines Workflow-Management-Systems um Transaktionskonzepte

Um transaktionale Workflows zu realisieren, wird das im vorherigen Abschnitt beschriebene WfMS um eine Transaktionssteuerung erweitert. Dadurch können dann mehrere Workflowaktivitäten zu einer Transaktion zusammengefasst werden. Die Steuerung des transaktionalen Workflows übernimmt weiterhin die Workflow-Engine. Änderungen werden dabei nicht sofort durchgeführt, sondern erst einmal nur vorgemerkt. Am Ende der Transaktion weist die Workflow-Engine die Transaktionssteuerung an, alle Änderungen durchzuführen und sichtbar zu machen. Damit werden die ACID-Eigenschaften garantiert.

Als Transaktionssteuerung wird der von der OMG spezifizierte Object Transaction Service (OTS) [OMG03] verwendet, welcher wie die OMG Workflowspezifikation, auf CORBA basiert. OTS unterstützt verteilte Transaktionen, kennt *Nested* und *Flat Transactions* und kann gut mit Systemen zusammenarbeiten, welche Transaktionen auf Grundlage des Distributed Transaction Processing (DTP) Protokolls [DTP96] ausführen. DTP wird von den meisten Datenbanken unterstützt. Durch die Verwendung von OTS können Datenbankressourcen sehr einfach in einen transaktionalen Workflow integriert werden.

Um die Interoperabilität mit anderen WfMS nicht zu verlieren, sollen die Komponenten aus dem WfMC-Referenzmodell möglichst nicht verändert, sondern nur erweitert werden. Damit das WfMS mit dem Transaktionsservice zusammen arbeiten kann, müssen zwischen beiden Systemen entsprechende Schnittstellen geschaffen werden. Die Zusammenführung wird dadurch erleichtert, dass beide Modelle auf CORBA basieren.

Abbildung 5 zeigt die Schnittstellen des Workflow- und Transaktionssystems. Dazwischen wurden neue Schnittstellen eingefügt, welche beide Systeme miteinander verbinden.

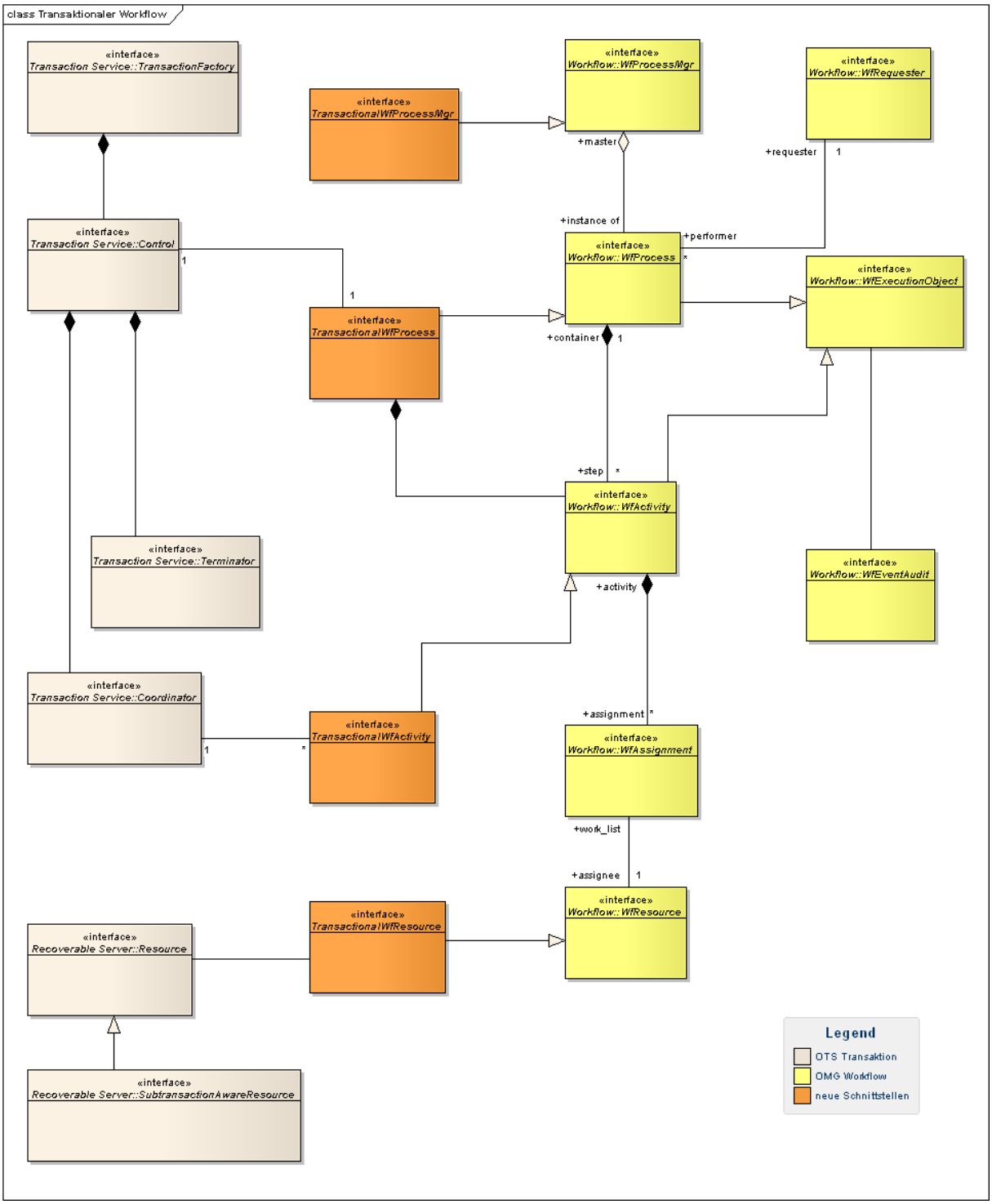


**Abbildung 5:** Schnittstellen des transaktionalen Workflows

In Abbildung 6 sind die Komponenten eines transaktionalen WfMS zu sehen. Es wird gezeigt, welche Schnittstellen die Komponenten besitzen und wie diese miteinander verbunden sind.

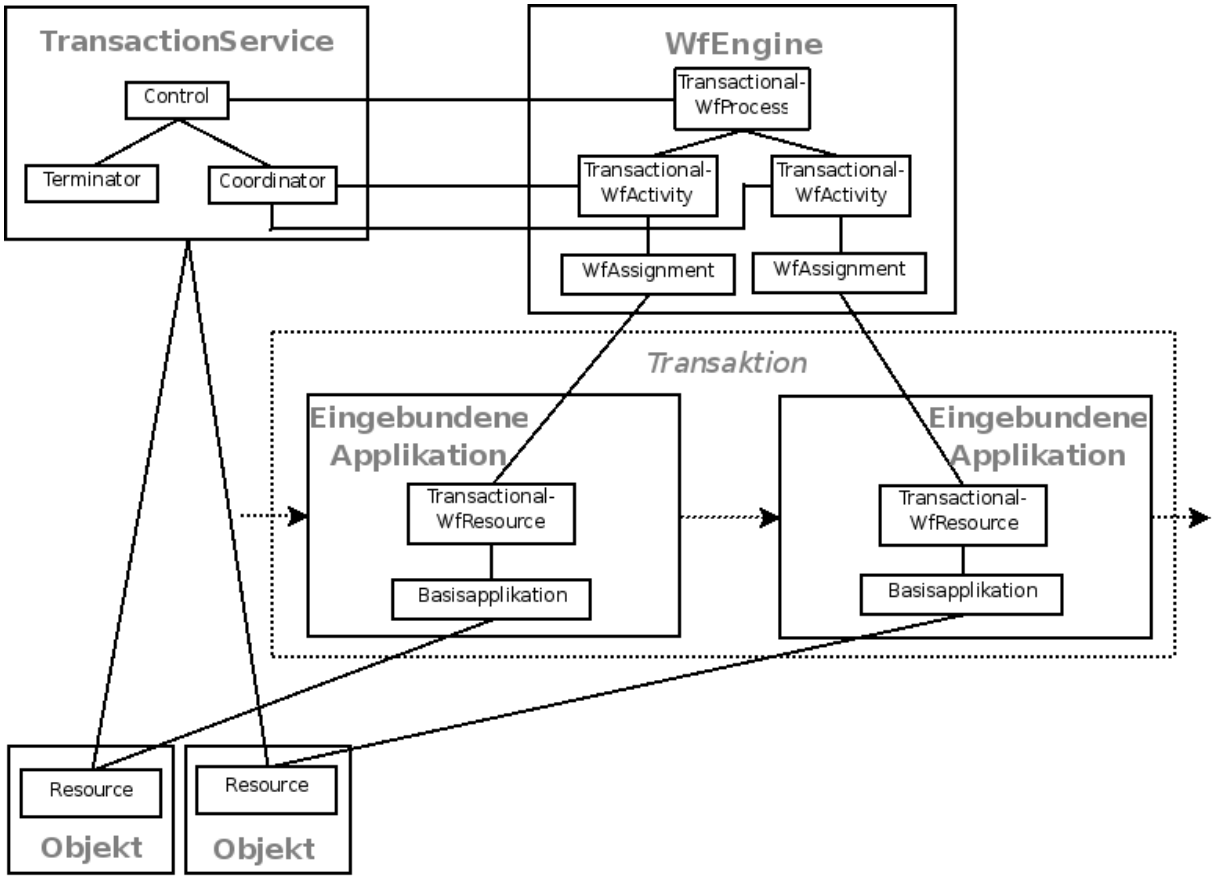


**Abbildung 6:** Komponenten des transaktionalen Workflows

Die Schnittstellen haben dabei folgende Aufgaben:

**TransactionalWfProcess:** Hierbei handelt es sich um die Instanz eines transaktionalen Workflowprozesses. Der gesamte Prozess wird als Transaktion ausgeführt.

**TransactionalWfActivity:** Die TransactionalWfActivity ist, wie WfActivity, die interne Repräsentation einer Workflowaktivität. Im Gegensatz zur WfActivity wird die Aktivität allerdings als Teiloperation einer Transaktion ausgeführt.

**TransactionalWfResource:** Die TransactionalWfResource stellt die Schnittstelle zu einer Applikation dar, welche als Teil eines transaktionalen Workflows ausgeführt werden soll.

**Resource:** Diese Schnittstelle stellt eine Ressource dar, welche während einer Transaktion von einer Applikation verändert wird. Die Änderungen werden allerdings nicht sofort übernommen, sondern erst einmal nur vorgemerkt. Erst am Ende der Transaktion wird ein Commit-Protokoll ausgeführt und alle veränderten Ressourcen werden aktualisiert.

**Control:** Die Control Schnittstelle stellt eine Transaktionsinstanz dar. Sie kennt einen Coordinator und einen Terminator.

**Coordinator:** Der Coordinator bietet Methoden an, um Resource-Objekte zu einer Transakti-

onsinstanz hinzuzufügen, den Transaktionsstatus abzufragen oder Subtransaktionen zu starten, welche der aktuellen Instanz untergeordnet werden.

**Terminator:** Der Terminator wird vom TransactionalWfProcess am Ende der Workflowtransaktion aufgerufen. Der Terminator sorgt dafür, dass das Commit-Protokoll mit allen registrierten Ressource-Objekten durchgeführt wird.

# 6 Ideen zur praktischen Umsetzung

Zur praktischen Umsetzung eines WfMS wird neben der in Kapitel 5 beschriebenen Architektur auch eine Workflowbeschreibung benötigt. Um Workflows mit Transaktionssemantik zu beschreiben, wurde von uns XPDL 2.0 verwendet. In der Spezifikation der Sprache existieren bereits Mechanismen, welche es erlauben, einen Sub-Workflow als Transaktion zu markieren. Um auch OTS Transaktionen und transaktionale CORBA-Aktivitäten damit beschreiben zu können, wurden entsprechende Erweiterungen in die Beschreibungssprache eingefügt.

Um die nachweisrelevanten Operationen und die Protokollierung als Transaktion in den Workflow einfügen zu können, müssen diese transaktional arbeiten. Durchgeführten Änderungen dürfen nicht sofort angewendet werden, sondern müssen vorerst nur im Transaktionskontext geschehen. Das bedeutet, Aktivitäten müssen Ressourcen implementieren, welche an einem Commit-Protokoll teilnehmen können. Dabei ist zu beachten, dass die nachweisrelevante Operation und die Protokollierung erst beim Commit-Befehl und nicht schon in einer Vorbereitungsphase des Commit-Protokolls durchgeführt werden.

Im Rahmen dieser Arbeit ist ein Prototyp entstanden, welcher das Architekturmodell eines transaktionalen Workflows aus Kapitel 5 umsetzt. Der Prototyp hat gezeigt, dass eine Umsetzung der beschriebenen Architektur prinzipiell möglich ist. Für die Zukunft ist geplant, dieses Architekturmodell weiter auszubauen und ein darauf basierendes WfMS zu entwickeln.

# 7 Zusammenfassung und Ausblick

In der vorliegenden Arbeit haben wir gezeigt, wie auf Basis des Workflow-Referenz-Modells der WfMC ein Workflow-Management-System zur verbindlichen Nachweisführung prinzipiell aufgebaut sein muss. Dazu wurde das Modell um den OTS Transaktionsservice erweitert. Durch die Verwendung der von der WfMC spezifizierten Schnittstellen und der Beschreibungssprache XPDL wird zudem die Interoperabilität mit existierenden Systemen gewährleistet. Ein entwickelter Prototyp hat gezeigt, dass die vorgestellte Architektur praktisch realisierbar ist.

Die Ergebnisse dieser Arbeit bilden die Grundlage für ein Lösungskonzept zur sichere Nachweisführung bei der Ausführung von IT-Operationen. Eine der großen Schwächen in der gegenwärtigen Nutzung von IT-Lösungen und der damit verbundenen Nachweisführung besteht in der Verwendung von Identitäten, die keinerlei oder nur wenig Vertrauenswürdigkeit aufweisen.

secunet erarbeitet zur Zeit Sicherheitskomponenten und Lösungen, die dafür sorgen, dass die Ausführung von Operationen auf vertrauenswürdige Identitäten zurückgeführt und mit diesen fest verknüpft werden können. Mit speziellen Methoden der Authentifikation und anderen technischen Rahmenbedingungen sind die Aktivitäten mit einer Art Willensbekundung des Akteurs verknüpft.

Mit den Mitteln zur transaktionalen Steuerung werden bestimmte Aktivitäten von Akteuren

in eine neue Qualität der Nachweisführung überführt. So wie die verwendeten Identitätsdaten müssen auch die Protokolldaten eine hohe Vertrauenswürdigkeit aufweisen, um im Zweifelsfall eine Nicht-Abstreitbarkeit der ausgeführten Operation zu garantieren.

# Literatur

[BPE07] *Web Services Business Process Execution Language Version 2.0*. OASIS Standard, April 2007.

[BPM06] *Business Process Modeling Notation Specificaton*. OMG Specification, Februar 2006.

[DTP96] *Distributed Transaction Processing: Reference Model, Version 3*. Technical Standard, Februar 1996.

[GMS87] GARCIA-MOLINA, HECTOR und KENNETH SALEM: *Sagas*. In: *SIGMOD '87: Proceedings of the 1987 ACM SIGMOD international conference on Management of data*, Seiten 249–259, New York, NY, USA, 1987. ACM Press.

[GR92] GRAY, JIM und ANDREAS REUTER: *Transaction Processing: Concepts and Techniques*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1992.

[Hol95] HOLLINGSWORTH, DAVID: *The Workflow Reference Model Version 1.1*. The Workflow Management Coalition Specification, Januar 1995.

[IBM] *IBM WebSphere Process Server*. Webseite: http://www-306.ibm.com/software/integration/wps/.

[Mar08] MARK, STEFAN: *Entwurf und prototypische Implementierung einer transaktionsorientierten Steuerung für Workflows*. Diplomarbeit, Technische Universität Dresden, 2008.

[Mos81] MOSS, J. ELIOT B.: *Nested Transactions: An Approach to Reliable Distributed Computing*. Doktorarbeit, Massachusetts Institute of Technology, 1981.

[OMG00] *Workflow Management Facility, V1.2*. OMG Specification, Mai 2000.

[OMG03] *Transaction Service Specification v1.4*. OMG Specification, September 2003.

[RS95] REUTER, ANDREAS und FRIEDEMANN SCHWENKREIS: *ConTracts - A Low-Level Mechanism for Building General-Purpose Workflow Management-Systems*. Data Engineering Bulletin, 18(1):4–10, 1995.

[WMC] *WfMC Standards and Working Groups*. Webseite: http://wfmc.org/standards/publicdocuments.htm.

[WMC99] *Workflow Management Coalition Terminology & Glossary*. The Workflow Management Coalition Specification, Februar 1999.

[XPD05] *Workflow Process Definition Interface - XML Process Definition Language V2.0*. The Workflow Management Coalition Workflow Standard, 2005.