WEBFLOW: Ein System zur flexiblen Ausführung webbasierter, kooperativer Workflows

Ulrike Greiner Erhard Rahm

Institut für Informatik, Universität Leipzig
http://dbs.uni-leipzig.de

Abstract: Der zunehmende Einsatz von Web-Technologien steigert die Notwendigkeit, kooperative Geschäftsprozesse und Workflows zwischen verschiedenen Unternehmen oder Abteilungen webbasiert abzuwickeln. Eine treibende Kraft hierfür ist die wachsende Verfügbarkeit von Web-Service-Realisierungen zur Interoperabilität. Die derzeit verfügbaren Ansätze unterstützen jedoch keine ausreichende Qualität und Flexibilität kooperativer webbasierter Workflows, insbesondere bzgl. der Ausführungsqualität von Diensten und der dynamischen Ausnahmebehandlung. Diese Defizite sollen mit dem System WEBFLOW abgebaut werden. Es ermöglicht die Definition und Überwachung von Ausführungsbedingungen für unterschiedlich mächtige Dienste verschiedener Kooperationspartner. Zudem unterstützt es eine regelbasierte dynamische Ausnahmebehandlung, mit der die Robustheit kooperativer Workflows deutlich verbessert werden soll.

1 Einleitung

Workflow-Management-Systeme unterstützen die Definition und Ausführung sich oft wiederholender und in ihrer Struktur weitgehend gleichbleibender Geschäftsprozesse und haben in Unternehmen, Kliniken oder Verwaltungen weite Verbreitung gefunden [GHS95], [JB96], [La97]. Diese Geschäftsprozesse und Workflows sind zunehmend kooperativ zwischen Unternehmen oder verschiedenen, auch geografisch verteilten Abteilungen abzuwickeln, die typischerweise unterschiedlichste Workflow-Systeme, Anwendungssysteme und Datenbanken einsetzen. Durch die weitgehende Web-Anbindung von Unternehmen und Einrichtungen sowie die zunehmende Verfügbarkeit von Web-Service-Implementierungen besteht nun erstmals die Möglichkeit, in größerem Umfang *kooperative Workflows* zu realisieren [LRS02]. Darunter verstehen wir Workflows, deren Aktivitäten durch Dienste unterschiedlicher Organisationseinheiten ausgeführt werden, wie die Bearbeitung eines Bauantrags, die verteilte Konstruktion von Fahrzeugen oder die Behandlung eines Patienten durch verschiedene niedergelassene Ärzte und Kliniken.

Derzeitige Web-Service-Realisierungen und -Standards decken primär die Sicherstellung einer Basis-Interoperabilität auf Ebene der Nachrichtenprotokolle und Austauschformate und zugehörige Aufgaben wie Registrierung von Diensten ab. Web Services werden derzeit auch primär zur Realisierung einfacher Dienste mit relativ kurzer Ausführungszeit verwendet, z.B. zur Ausführung einer einfachen Anwendungsfunktion oder einer Datenbankabfrage. Kooperative Workflows erfordern jedoch die Zusammenarbeit zwischen

komplexeren Diensten, die selbst durch einen Workflow realisiert sein können und durch oft lange Ausführungsdauer, Nutzerinteraktionen etc. gekennzeichnet sind. Besondere Herausforderungen bei der Realisierung der Workflow-Kooperation entstehen u.a. durch die Komplexität der einzubindenden Dienste, die Autonomie der beteiligten Unternehmen z.B. bzgl. der Implementierung ihrer Dienste sowie durch die für eine breite Nutzerakzeptanz notwendige Robustheit kooperativer Workflows. Deshalb ist beispielsweise eine ausreichende Flexibilität, insbesondere zur Behandlung von Fehlern und Ausnahmen, von größter Bedeutung. Neben erweiterten Recovery-Verfahren mit der Möglichkeit kompensationsbasierter Zurücksetzung von Teilschritten aufgrund von Systemausfällen und sonstigen technischen Fehlern werden flexible Mechanismen zur Behandlung logischer Ausnahmesituationen (Produktangebot überschreitet Preislimit, Liefertermin wird überschritten, etc.) benötigt. Bei diesen soll ein betroffener kooperativer Workflow nach der notwendigen Ausnahmebehandlung weiter „nach vorne“ ausgeführt werden können. Weiterhin sollten möglichst viele der Ausnahmen automatisch erkannt und behandelt werden können, um auch bei sehr häufiger Ausführung kooperativer Workflows eine hohe Robustheit und Ausführungsqualität zu erreichen.

Zur Umsetzung dieser Ziele entwickeln wir derzeit das webbasierte System WEBFLOW, dessen Grobentwurf in diesem Beitrag vorgestellt wird. WEBFLOW basiert auf folgenden Eigenschaften:

- Definition und Ausführung kooperativer Workflows im Rahmen einer Mediatorarchitektur, bei der heterogene Dienste über Organisationsgrenzen hinweg eingebunden werden und gleichzeitig die Autonomie der Kooperationspartner gewahrt bleibt;
- weitgehende Nutzung vorhandener Web-Service-Standards wie WSDL (Web Services Description Language, [Ch01]), UDDI (Universal Description, Discovery and Integration [Be02]) oder BPEL4WS (Business Process Execution Language for Web Services [Cu02]) und verfügbarer Implementierungen zur Gewährleistung der Interoperabilität und Reduzierung des Implementierungsaufwandes;
- Kooperationsmodell mit expliziter Spezifikation und Überwachung von temporalen und inhaltlichen Ausführungsbedingungen durch entsprechende Erweiterung von Diensten (Web Services). Die erweiterte Funktionalität wird durch den WEBFLOW-Mediator erbracht und kann sowohl für einfache Dienste (z.B. auf Basis von Legacy-Anwendungen) als auch für komplexe Dienste genutzt werden.
- Regelbasierte Erkennung und (teil-) automatische Behandlung von Ausnahmen mit flexiblen Reaktionsmöglichkeiten, insbesondere vorwärts orientierter Fortsetzung der Verarbeitung und dynamischer Adaption betroffener Workflows. Die Ausnahmebehandlung für einen Dienst kann für alle Workflows zentral im WEBFLOW-Mediator oder spezifisch in dem den Dienst verwendenden (kooperativen) Workflow festgelegt werden.

Im Folgenden werden nach einem Überblick über verwandte Arbeiten (Abschnitt 2) die Mediatorarchitektur in Abschnitt 3 und die Ausnahmebehandlung in Abschnitt 4 vorgestellt. Den Schluss bilden Zusammenfassung und Ausblick (Abschnitt 5).

2 Verwandte Arbeiten

In letzter Zeit beschäftigen sich verschiedene Forschungsarbeiten mit interorganisatorischen Workflows und der verteilten Ausführung von Workflows [AW01],[BRH02], [CS01],[Gr01],[La01],[Mu98],[SS01],[VW99]. Diese Ansätze bieten jedoch meist keine ausreichenden Möglichkeiten zur automatischen Überwachung von Ausführungsbedingungen bzw. unterstützen keine heterogenen Kooperationspartner mit unterschiedlich mächtigen Diensten. Eine Ausnahmebehandlung ist wenn überhaupt nur in einfacher Form vorhanden, typischerweise für die Überwachung von Deadlines. Einige in früheren Forschungsarbeiten vorgeschlagenen Konzepte zur dynamischen Ausnahmebehandlung sollen für die spezifischen Zielsetzungen von WEBFLOW angepasst werden, insbesondere die Verwendung einer regelbasierten Ausnahmeerkennung sowie dynamische Workflow-Adaptionen [MR00], [Re01].
Die Workflow Management Coalition hat mit Interface 4 eine Schnittstelle für die Kooperation verschiedener Workflow-Engines spezifiziert [Wf00], die sich aber auf den Datenaustausch konzentriert und keine Ausnahmebehandlungsmöglichkeiten enthält.
Web Services auf Basis von WSDL [Ch01] ermöglichen nur eine einfache Fehlerbehandlung durch vordefinierte Fehlernachrichten, unterstützen jedoch keine Ausführungsbedingungen. Die kürzlich erschienenen Standardisierungsvorschläge *BPEL4WS* (*Business Process Execution Language for Web Services* [Cu02]) und *WS-Transaction* [Ca02] zur Definition von Geschäftsprozessen auf der Basis von Web Services sehen eine Transaktionsunterstützung und eine kompensationsbasierte Ausnahmebehandlung beim Empfang vordefinierter Fehlernachrichten oder der Verletzung von Timeouts vor; ähnliches gilt für das Workflow-System Microsoft BizTalk Server [Me01], [Ro01]. Diese Ansätze können nur auf Ausnahmen reagieren, für die vordefinierte Fehlernachrichten existieren; zudem kann die Ausnahmebehandlung für einen Dienst nicht zentral festgelegt werden, sondern muss in jedem den Dienst verwendenden Workflow spezifiziert werden.

3 WEBFLOW-Architektur und Metadatenmodell

Wir beschreiben zunächst die Grobarchitektur von WEBFLOW und skizzieren danach die verwendeten Metadaten und das Kooperationsmodell.

3.1 Architekturüberblick

WEBFLOW basiert auf einer Mediatorarchitektur (Abb. 1), die analog zu Mediatoransätzen für die Integration heterogener Datenquellen die Integration heterogener Dienste verschiedener Organisationseinheiten in kooperative Workflows unter Wahrung der Autonomie der Anbieter unterstützt. Die zentrale Komponente, der WEBFLOW-Mediator, verfügt über die Funktionalität eines Workflow-Systems zur Ausführung kooperativer Workflows, in deren Rahmen unterschiedliche Dienste derselben Organisationseinheit oder externer Knoten eingebunden werden. Für diese Dienste können zur Verbesserung der Kooperationsqualität Ausführungsbedingungen definiert und überwacht werden; zur Verbesserung der Ausführungsflexibilität erfolgt eine (teil-) automatische Ausnahmebehandlung.
Die Ausführung der Dienste erfolgt typischerweise durch Anwendungssysteme und Da-

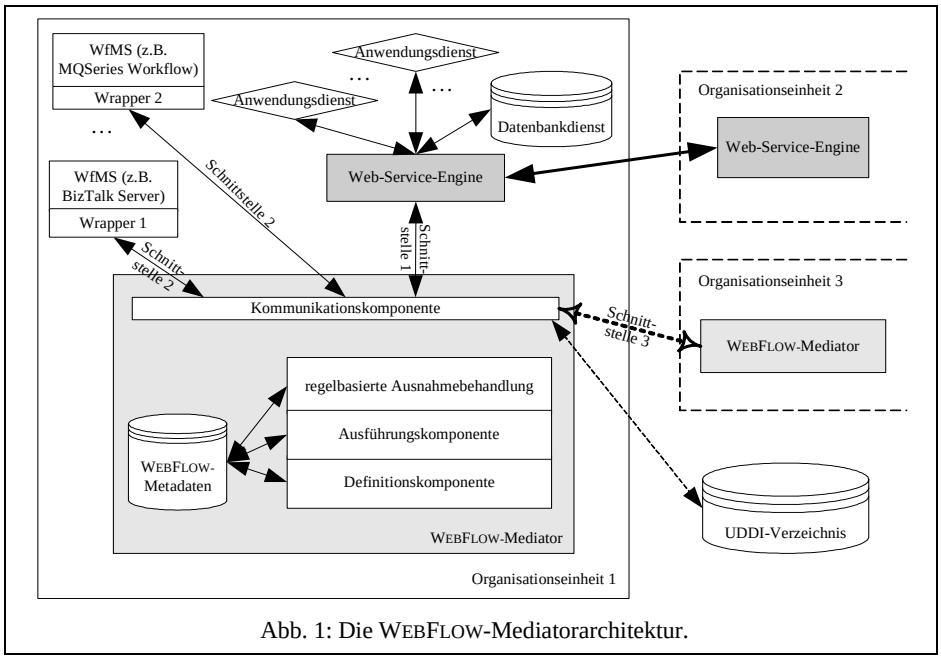

Abb. 1: Die WEBFLOW-Mediatorarchitektur.

tenbanksysteme, die vom Mediator über eine Web-Service-Engine angesprochen werden (Schnittstelle 1). Da Workflows derzeit meist nicht als Web Services definiert werden können, unterstützt WEBFLOW eine erweiterte WSDL-ähnliche Dienst-Schnittstelle für Workflow-Management-Systeme (WfMS) (Schnittstelle 2).

Wie Abb. 1 zeigt, besteht der WEBFLOW-Mediator aus fünf Hauptkomponenten:

- *Definitionskomponente* zur Definition von Diensten, Ausführungsbedingungen, Ausnahmeregeln und kooperativen Workflows. Einfache Dienste können dabei typischerweise aus UDDI-Verzeichnissen, komplexe Dienste von Mediatoren externer Kooperationspartner importiert werden.
- *Ausführungskomponente* zur Ausführung der kooperativen Workflows einer Organisationseinheit und zur Überwachung von Ausführungsbedingungen während der Workflow-Ausführung,
- *regelbasierte Ausnahmebehandlung*, die mit Hilfe spezieller ECA-Regeln die Ausnahmeerkennung und -behandlung durchführt,
- den *WEBFLOW-Metadaten* mit Informationen zu kooperativen Workflows, Diensten, Ausführungsbedingungen und Ausnahmen (s. 3.2),
- *Kommunikationskomponente*, die für den Mediator die unterschiedlichen Schnittstellen zu anderen Systemen verbirgt.

Kooperative Workflows werden ausgeführt, indem die Ausführungskomponente entsprechend des Kontrollflusses den oder die nächsten auszuführenden Dienste bestimmt und aufruft. Der Dienstaufruf kann sowohl synchron als auch asynchron erfolgen. Bei lokalen Diensten erfolgt der Aufruf über die lokale Web-Service-Engine (Schnittstelle 1) oder geht direkt an ein WfMS (Schnittstelle 2). Der Aufruf externer Dienste erfolgt entweder über die lokale Web-Service-Engine an die Web-Service-Engine des Partners oder direkt an

den WEBFLOW-Mediator des Partners (Schnittstelle 3). Über die Schnittstelle 3 werden unter anderem auch Metadaten zu Diensten externer Organisationseinheiten oder Nachrichten bei der Verletzung von Ausführungsbedingungen, die der Mediator des Partners überwacht, ausgetauscht (s. Abschnitt 4.1 zur Überwachung von Ausführungsbedingungen bei der Ausführung von Diensten).
Um eine große Flexibilität bzgl. der Ausnahmebehandlung zu erreichen, ist geplant als Workflow-Engine für WEBFLOW ADEPT$_{flex}$ [Re01] einzusetzen, das Operationen zur manuellen, dynamischen Workflow-Adaption anbietet. Außerdem werden in WEBFLOW Teile eines im Rahmen eines DFG-Projekts entwickelten Prototypen zur ereignisbasierten, dynamischen Workflow-Adaption verwendet (u.a. für die regelbasierte Ausnahmeerkennung).

3.2 WEBFLOW-Metadaten

Eine wichtige Komponente der WEBFLOW-Architektur sind die WEBFLOW-Metadaten, die unter anderem Informationen über kooperative Workflows, beteiligte Dienste, Ausführungsbedingungen, Ausnahmen und Regeln zur Ausnahmebehandlung enthalten. Sie werden z.B. bei der Überwachung von Ausführungsbedingungen oder für die regelbasierte Ausnahmebehandlung verwendet (Abschnitt 4). Abbildung 2 zeigt einen Ausschnitt des Metadaten-Modells in UML-Notation.
Die WEBFLOW-Metadaten umfassen alle notwendigen Informationen zu *kooperativen Workflows*, die aus *Diensten* bestehen, die von unterschiedlichen *Organisationseinheiten* ausgeführt werden. Dienste werden in *nicht-interaktive Dienste* (üblicherweise Web Services, die nach Aufruf ohne Nutzerinteraktion ausgeführt werden) und *interaktive Dienste*

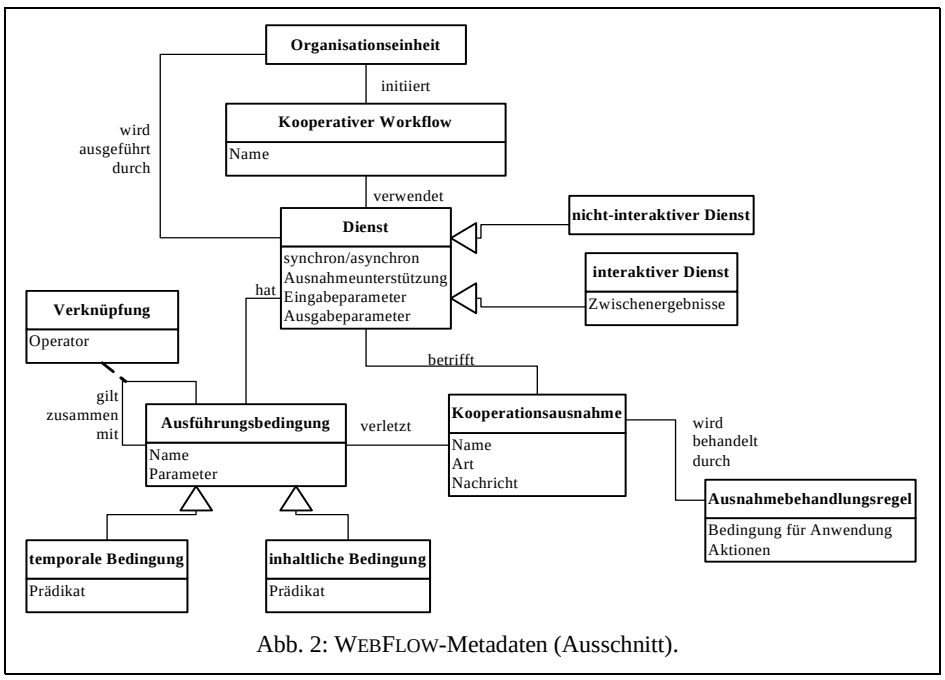

Abb. 2: WEBFLOW-Metadaten (Ausschnitt).

(typischerweise Workflows, die auch Zwischenergebnisse zurückgeben) unterteilt. Alle Arten von Diensten werden durch ihre *Aufrufart* (synchron/asynchron), ihre *Ein-* und *Ausgabeparameter* und den Grad der *Ausnahmeunterstützung* (siehe 3.3) charakterisiert.
Zur Qualitätskontrolle der Workflow-Kooperation können zu Diensten *Ausführungsbedingungen* definiert werden. Wir unterscheiden *temporale* Bedingungen wie Zeitpunkte („Auftrag muss bis 02.03.2003 bearbeitet werden“) oder Intervalle („Ergebnis muss innerhalb von 5 Tagen vorliegen“) und *inhaltliche* Bedingungen („Preis < 100 EUR“) oder Vorgaben („Gutachten muss Aussage zu xy enthalten“). Ausführungsbedingungen beziehen sich auf Ein- oder Ausgabeparameter eines Dienstes (Attribut *Parameter*) und spezifizieren einen zulässigen Wertebereich. Hat ein Dienst mehrere Ausführungsbedingungen, so werden diese mit booleschen Operatoren verknüpft (Assoziation *gilt zusammen mit*, Klasse *Verknüpfung*). Die Abbildung der Ausführungsbedingungen in einer eigenen Klasse erleichtert ihre Überwachung durch die erweiterte Ausführungskomponente des WEBFLOW-Mediators. Außerdem entsteht durch die Ausführungsbedingungen eine Klassifikation der Dienste, die z.B. die Suche nach bestimmten Diensten bei der Definition kooperativer Workflows erleichtert (z.B. „Finde einen Dienst, der den Auftrag innerhalb drei Tagen bearbeiten kann“).
Die für die Ausführungsflexibilität wesentliche Ausnahmebehandlung erfordert die automatische Erkennung und Behandlung von *Kooperationsausnahmen*. Die Kooperationsausnahmen bilden den Event-Teil der ECA-Regeln für die regelbasierte Ausnahmebehandlung. Condition- und Action-Teil stehen in der Klasse *Ausnahmebehandlungsregel*, die eine *Bedingung* für die Anwendung der Regel sowie die notwendigen *Aktionen* enthält. Abbildung 3 zeigt eine solche ECA-Regel für die Ausnahme, dass ein Fertigstellungstermin für ein Gutachten nicht eingehalten werden kann (Ereignis) und das Gutachten dringend benötigt wird (Bedingung, zu den Aktionen s. Abschnitt 4.2).

EREIGNIS	Terminüberschreitung durch Sachverständigen
BEDINGUNG	hohe Dringlichkeit des Auftrags
AKTIONEN	Suche alternativen Sachverständigen. Falls ein solcher existiert: Abbruchnachricht an verspäteten Dienst. Passe Fristen in den betroffenen kooperativen Workflows an.

Abb. 3: Beispielregel zur Ausnahmebehandlung.

Zu einer Kooperationsausnahme kann es mehrere ECA-Regeln geben, um die Ausnahmebehandlung in Abhängigkeit von konkreten Laufzeitdaten zu spezifizieren. Durch die Definition der Ausnahmebehandlung auf der Ebene der Metadaten müssen die Regeln für jeden Dienst nur einmal definiert werden und nicht redundant in allen kooperativen Workflows enthalten sein, die diesen verwenden. Damit kann insbesondere eine vorwärts orientierte Ausnahmebehandlung unterstützt werden, außerdem wird die Übersichtlichkeit der kooperativen Workflows erhöht.

3.3 Ausnahmeunterstützung

Wie bereits im vorigen Abschnitt dargestellt werden Dienste, die in einem kooperativen

Workflow verwendet werden, u.a. durch den Grad der sogenannten *Ausnahmeunterstützung* charakterisiert. Diese beschreibt welche Informationen über den Aufruf und die Ein- und Ausgabeparameter hinaus zwischen Diensten und dem WEBFLOW-Mediator fließen und für die Ausnahmeerkennung genutzt werden können. Wir unterscheiden drei Grade:

- Grad 0 - keine Ausnahmeunterstützung:
  keine Fehlernachrichten, keine Ausführungsbedingungen
- Grad 1 - Standard-Ausnahmeunterstützung:
  Fehlernachrichten bei erfolgloser Ausführung, keine Ausführungsbedingungen
- Grad 2 - erweiterte Ausnahmeunterstützung:
  Fehlernachrichten und temporale oder inhaltliche Ausführungsbedingungen

Grad 0 gilt für Dienste, die keinerlei Fehlermeldungen schicken, z.B. einfache Applikationen, die in Web Services gekapselt wurden. Grad 1 beschreibt Dienste, die vordefinierte Fehlernachrichten bei Ausführungsfehlern verschicken (z.B. „unvollständigen Daten für die Bearbeitung einer Anfrage“, „Artikel vergriffen“). Solche Fehlernachrichten sind z.B. die fault-Nachrichten einer WSDL-Web-Service-Definition. Unterstützt ein Dienst zusätzlich temporale oder inhaltliche Ausführungsbedingungen entspricht er Grad 2.
Die obige Klassifikation ist orthogonal zur Einteilung in nicht-interaktive und interaktive Dienste in Abschnitt 3.2; derzeit sind jedoch nicht-interaktive Dienste, die über eine Standard-Web-Service-Schnittstelle aufgerufen werden, typischerweise vom Grad 0 oder 1. Grad 2 hingegen ist überwiegend bei interaktiven Diensten von Bedeutung, da hier eine erweiterte Schnittstelle und zusätzliche Funktionalität beim Anbieter zur Überwachung der Ausführungsbedingungen erforderlich sind. Dadurch können Ereignisse, die voraussichtlich zu einer Verletzung von Ausführungsbedingungen führen (wie die Erkrankung eines Sachverständigen), an den WEBFLOW-Mediator gemeldet werden *bevor* die eigentliche Verletzung eintritt. Zur Erhöhung der Kooperationsqualität sind also Dienste vom Grad 2 wünschenswert. Da aber derzeit nur wenige Dienste diesen Grad unterstützen, bietet WEBFLOW die Möglichkeit, Ausführungsbedingungen auch für Dienste mit Ausnahmeunterstützung vom Grad 0 oder 1 zu definieren und zu überwachen ohne in die Anwendungen, die die Dienste ausführen, einzugreifen, was insbesondere bei Diensten externer Organisationseinheiten von Vorteil ist.

4 Dynamische Ausnahmebehandlung

Die dynamische Ausnahmebehandlung von WEBFLOW gliedert sich in die Erkennung einer Kooperationsausnahme und die Reaktion auf die Ausnahme. Ziel ist es, die Ausführung eines kooperativen Workflows auch nach Kooperationsausnahmen möglichst „nach vorne“ fortzusetzen. Außerdem sollen von der Ausnahme betroffene Dienste frühzeitig über zu erwartende Verzögerungen oder Probleme informiert werden. Die zuverlässige Ausführung verteilter Workflows auf Basis transaktionaler Dienste und verteilter Commit-Protokolle ist orthogonal zur dynamischen Ausnahmebehandlung und wird hier nicht weiter betrachtet; hierzu sollen an *WS-Transaction* [Ca02] angelehnte Realisierungen verwendet werden.

4.1 Ausnahmeerkennung

Welche Schritte zur *Erkennung* einer Ausnahme nötig sind, wird von der Art der Ausnahme und vom Grad der Ausnahmeunterstützung eines Dienstes (vgl. 3.3) bestimmt. Ausnahmen entstehen entweder durch Ausführungsfehler eines Dienstes, durch die Verletzung von Ausführungsbedingungen oder durch andere, unerwartete Ereignisse, die die erfolgreiche Dienstausführung beeinflussen (wie z.B. Krankheit eines Sachverständigen). Tabelle 1 gibt einen Überblick, welche Möglichkeiten bei unterschiedlicher Ausnahmeunterstützung bestehen Ausführungsfehler und Bedingungsverletzungen zu erkennen.
Bei Diensten mit Ausnahmeunterstützung vom Grad 0, also ohne Fehlernachrichten oder Ausführungsbedingungen, können Ausnahmen aufgrund von Ausführungsfehlern (z.B. unkorrekte Ausführung, Absturz der Anwendung, Netzfehler, Überlastung) entweder daran erkannt werden, dass ein aufgerufener Dienst nicht innerhalb einer vorgegebenen Zeit antwortet oder Rückgabeparameter sinnlose Werte enthalten. Die Erkennung erfolgt also durch beim Dienstaufruf gesetzte Timeouts oder eine semantische Prüfung der Rückgabeparameter. Bei Standard- und erweiterter Ausnahmeunterstützung werden Ausführungsfehler durch vordefinierte Fehlernachrichten angezeigt (z.B. „Artikel nicht verfügbar“).
Die Verletzung von Ausführungsbedingungen wird bei Diensten vom Grad 0 und 1 durch die Überwachung der Bedingungen durch den Mediator erkannt. Temporale Bedingungen werden dabei mit Hilfe von Timeouts oder Deadlines und inhaltliche Bedingungen durch die Überprüfung der zurückgegebenen Ausgabeparameter überwacht (z.B. „Ist der Preis größer als das vorgegebene Limit?“). Damit werden Verletzungen erst erkannt, wenn der gesetzte Timeout verstrichen ist oder die Dienstausführung beendet und die Parameter zurückgegeben wurden. Dienste vom Grad 2, die ihre Ausführungsbedingungen selbst überwachen, schicken bei einer Verletzung Fehlernachrichten an den Mediator.
Ausnahmen der dritten Art, unerwartete Ereignisse, die die erfolgreiche Dienstausführung behindern, können nur bei interaktiven Diensten erkannt werden, da sie durch einen Nutzer z.B. in Form einer E-Mail oder eines Telefonanrufs gemeldet werden müssen; anschließend werden sie über eine Eingabemaske an den WEBFLOW-Mediator weitergeleitet.

4.2 Reaktion

WEBFLOW verwendet zur Ausnahmebehandlung eine mehrstufige Vorgehensweise. Wurde eine Kooperationsausnahme erkannt, werden zunächst anhand der in den Metadaten vorhandenen Informationen automatisch alle potenziell von einer Ausnahme *betroffenen kooperativen Workflow-Instanzen* bestimmt. Im Beispiel aus Abb. 3 könnten das alle Instanzen sein, die Gutachten hoher Dringlichkeit bei dem gleichen Sachverständigen in Auftrag gegeben haben und deren Fertigstellungstermine (die z.B. in einer Ausführungs-

	Ausführungsfehler	Bedingungsverletzung
Grad 0 - keine	Timeouts / semantische Prüfung	Überwachung von Ausführungsbedingungen
Grad 1 - Standard	vordefinierte Fehlernachrichten	
Grad 2 - erweitert	vordefinierte Fehlernachrichten	Fehlernachrichten

Tab. 1: Erkennung von Ausnahmen in Abhängigkeit von der Ausnahmeunterstützung.

bedingung abgelegt sind) innerhalb der nächsten 24 Stunden liegen.
Danach wird in den Metadaten eine passende *Regel* mit den notwendigen Aktionen gesucht. Wird keine passende Regel gefunden, wird geprüft, ob in der betroffenen Workflow-Instanz eine Aktion definiert ist und gegebenenfalls zur Anwendung gebracht. Wurde auch dort nichts spezifiziert, erfolgt die Benachrichtigung eines verantwortlichen Nutzer, so dass dieser manuell auf die Ausnahme reagieren kann.

Die möglichen Reaktionen auf eine Ausnahme, die im Aktionsteil der Regeln festgelegt sind, hängen stark vom verwendeten Workflow-Modell und den Fähigkeiten der verwendeten Workflow-Engine ab:

- Bei den meisten kommerziellen Workflow-Systemen kann man den Workflow nach Auftreten einer Ausnahme nur abbrechen und/oder eine manuelle Ausnahmebehandlung durchführen.
- Manche Systeme (z.B. Microsofts BizTalk Server) bieten die Möglichkeit Kompensationsprozesse zu definieren, die beispielsweise einen Dienst erneut aufrufen oder einen alternativen Dienst ausführen, wenn eine Dienstausführung fehlschlägt.
- Ein weiterer Schritt zur Erhöhung der Flexibilität ist eine manuelle oder automatische Workflow-Adaption, bei der z.B. Aktivitäten gelöscht oder eingefügt oder Zeitkanten, die den zeitlichen Abstand zwischen zwei Aktivitäten festlegen, bei Verzögerungen angepasst werden (vgl. [MR00]). Um solche Änderungen zur Laufzeit durchzuführen, benötigt man eine Workflow-Engine wie $\text{ADEPT}_{\text{flex}}$ [Re01], die dynamische Änderungen unterstützt.

WEBFLOW verwendet ein Workflow-Modell, mit dem z.B. Zeitabhängigkeiten modelliert werden können und eine Workflow-Engine, die dynamische Änderungen unterstützt. Die möglichen Reaktionen auf die in Abb. 3 dargestellte Ausnahme umfassen demzufolge die Suche nach einem alternativen Sachverständigen, eine Abbruchnachricht an den verspäteten Sachverständigen, falls ein alternativer Dienst gefunden wurde, sowie die Anpassung von Fristen in den betroffenen kooperativen Workflows. Mit Hilfe der ECA-Regeln und der übrigen Metadaten können die notwendigen Aktionen durch die Ausnahmebehandlungskomponente des WEBFLOW-Mediators automatisch bestimmt und ausgeführt werden. Die Ausführung kann zur Sicherheit von der Bestätigung eines verantwortlichen Nutzers abhängig gemacht werden.

5 Zusammenfassung und Ausblick

Im vorliegenden Beitrag wurden Grobarchitektur, Kooperationsmodell und Ausnahmebehandlung des Systems WEBFLOW überblicksartig vorgestellt. Wesentliche Zielsetzung ist auf Basis von Web Services die Qualität, Flexibilität und Robustheit webbasierter kooperativer Workflows zu erhöhen, insbesondere durch die Definition und Überwachung von Ausführungsbedingungen für unterschiedlich mächtige Dienste und durch eine regelbasierte, teilautomatische und vorwärts orientierte Ausnahmebehandlung.
Zur Umsetzung der vorgestellten Konzeption wird derzeit die Definitionssprache für kooperative Workflows mit Ausführungsbedingungen näher spezifiziert, die sich an dem Workflow-Modell von $\text{ADEPT}_{\text{flex}}$ und [MR00] sowie an BPEL4WS orientiert. Zur WEBFLOW-Realisierung wird wie schon erwähnt auf einen im Rahmen eines DFG-Projekts ent-

wickelten Prototypen zur ereignisbasierten, dynamischen Workflow-Adaption zurückgegriffen, der unter anderem um die Komponenten zur Überwachung der Ausführungsbedingungen sowie zur Web-Service-Anbindung erweitert wird.

Danksagung: Wir bedanken uns für die wertvollen Hinweise der anonymen Gutachter. Die Arbeit wurde durch die Deutsche Forschungsgemeinschaft (DFG) unterstützt, Projektnummer Ra497-12.

Literaturverzeichnis

[AW01] van der Aalst, W.M.P., Weske, M.: The P2P Approach to Interorganizational Workflows. In Proc. of CAiSE 2001, S. 140-156.

[Be02] Bellwood, T. et al.: UDDI Version 3.0, Juli 2002. http://uddi.org/pubs/uddi_v3.htm

[BRH02] Bon, M.; Ritter, N.; Härder, T.: Sharing Product Data among Heterogenous Workflow Environments. In Proc. of CAD 2002, Dresden, 2002, S.139-150.

[Ca02] Cabrera, F. et al.: Web Services Transaction (WS-Transaction), August 2002. http://www-106.ibm.com/developerworks/library/ws-transpec/

[CS01] Casati, F.; Shan, M.-C.: Dynamic and adaptive composition of e-services. Information Systems 26, 2001, S. 143-163.

[Ch01] Christensen, E. et al.: Web Services Description Language (WSDL) 1.1, W3C Note, March 2001. http://www.w3.org/TR/wsdl

[Cu02] Curbera, F. et al.: Business Process Execution Language for Web Services, Version 1.0 Juli 2002. http://www-106.ibm.com/developerworks/library/ws-bpel/

[GHS95] Georgakopoulos, D.; Hornick, M.; Sheth, A.: An Overview of Workflow Management: From Process Modeling to Infrastructure for Automation. Journal on Distributed and Parallel Database Systems 3(2), 1995; S. 119-153.

[Gr01] Grefen, P.; Aberer, K.; Ludwig, H.; Hoffner, Y.: CrossFlow: Cross-Organiziational Workflow Management for Service Outsourcing in Dynamic Virtual Enterprises. In IEEE Bull. of the Tech. Comm. on Data Engineering, 24(1), March 2001, S. 52-57.

[JB96] Jablonski, S.; Bussler, C.: Workflow Management. Modeling Concepts, Architecture, and Implementation. International Thomson Computer Press, London, 1996.

[La97] Lawrence, P.: Workflow Handbook. John Wiley and Sons, New York, 1997.

[La01] Lazcano, A. et al.: WISE: Process based E-Commerce. In IEEE Bulletin of the Technical Committee on Data Engineering, Vol. 24, No.1, March 2001, S. 46-51.

[LRS02] Leymann, F.; Roller, D.; Schmidt M.-T.: Web services and business process management. In IBM Systems Journal, Vol. 41, No. 2, 2002, S. 198-211.

[Me01] Mehta, B. et al.: BizTalk Server 2000 Business Process Orchestration. In IEEE Bulletin of the Tech. Committee on Data Engineering, Vol. 24, No. 1, March 2001, S. 35-39.

[MR00] Müller, R.; Rahm, E.: Dealing with Logical Failures for Collaborating Workflows. In Etzion, O.; Scheuermann, P. (Hrsg.): Proc. of CoopIS 2000, Eilat, 2000. LNCS 1901, Springer: S. 210-233.

[Mu98] Muth, P. et al.: Enterprise-wide workflow management based on state and activity charts. In Dogac, A. et al. (Hrsg.): Workflow management systems and interoperability. NATO Advanced Study Institute, Springer, New York, 1998, S. 281-303.

[Re01] Reichert, M. et al.: Realisierung flexibler, unternehmensweiter Workflow-Anwendungen mit ADEPT. In P. Horster (Hrsg.): Proc. Elektr. Geschäftsprozesse - Grundlagen, Sicherheitsaspekte, Realisierungen, Anwendungen. Klagenfurt, 2001, S. 217-228.

[Ro01] Roxburgh, U.: BizTalk Orchestration: Transactions, Exceptions and Debugging. Microsoft White Paper, 2001.

[SS01] Schönhoff, M.; Stormer, H.: Trading Workflows Electronically: The ANAISOFT Architecture. In Proc. der BTW 2001, Oldenburg, 2001, S. 67-74.

[VW99] Vossen, G.; Weske, M.: The WASA2 Object-Oriented Workflow Management System. In Proc. ACM SIGMOD Conference, Philadelphia, 1999, S. 587-589.

[Wf00] Workflow Management Coalition: Workflow Standard-Interoperability. Wf-XML Binding. Doc. Nr. WFMC-TC-1023, 2000. http://www.wfmc.org