Thomas Bauer, Manfred Reichert

Dynamische Änderung von Serverzuordnungen in verteilten Workflow-Management-Systemen

Prozessorientierte Anwendungen lassen sich nur durch die Verwendung eines Workflow-Management-Systems (WfMS) mit vernünftigem Aufwand und zu akzeptablen Kosten realisieren. Jedoch stoßen zentrale WfMS mit nur einem einzigen Server zur Steuerung aller Workflows (WF) oftmals sehr schnell an ihre Leistungsgrenze. Zur Erhöhung der Skalierbarkeit von WfMS werden deshalb zahlreiche Ansätze für die Realisierung von WfMS mit mehreren WF-Servern vorgeschlagen. Bei vielen dieser Multi-Server-WfMS wird der konkrete WF-Server zur Kontrolle einer bestimmten WF-Aktivität meist durch eine Serverzuordnung festgelegt. Jedoch entstehen bei solchen verteilten WfMS Probleme, wenn einzelne Komponenten (WF-Server, Teilnetze oder Gateways) überlastet sind oder ausfallen. Eine aus anderen Bereichen der Informatik bekannte Vorgehensweise ist in solchen Fällen, die Hardwarezuordnung dynamisch zu verändern. In der vorliegenden Arbeit wird erstmalig untersucht, inwieweit durch eine dieser Vorgehensweise entsprechende dynamische Änderung von Serverzuordnungen in verteilten WfMS geeignet auf Überlast- und Ausfallsituationen reagiert werden kann.

1 Einleitung

WfMS ermöglichen die computerunterstützte Ausführung von Arbeitsprozessen (engl. Workflow; kurz: WF) in einer verteilten Systemumgebung (vgl. [AH02, JBS97, LR00, Obe96]). Der entscheidende Vorteil von WfMS ist, dass sie helfen, prozessorientierte Anwendungssysteme überschaubarer und damit besser wartbar zu gestalten. Dazu wird der applikationsspezifische Code der benutzten Anwendungen von der Ablauflogik des Geschäftsprozesses (»Prozesslogik«) getrennt. Anstelle eines großen monolithischen Programmpakets erhält man einzelne Aktivitäten, welche die Anwendungsprogramme repräsentieren. Ihre Ablauflogik wird in einer separaten Beschreibung festgelegt, welche die Ausführungsreihenfolge (Sequenz, bedingte oder parallele Verzweigung, Schleifen) der einzelnen Aktivitäten und die Datenflüsse zwischen ihnen bestimmt. Durch die beschriebene Trennung können Entwurfsfehler bzgl. des Prozessablaufs noch vor Implementierung der eigentlichen Anwendungskomponenten entdeckt werden. Aus denselben Gründen sind spätere Änderungen der Geschäftsprozesse und dadurch notwendige Anpassungen der Anwendungssysteme einfacher durchführbar.

Ausgehend von einem solchen WF-Modell können berechtigte Benutzer zur Laufzeit neue WF-Instanzen erzeugen und starten. Die Ausführung dieser WF-Instanzen wird vom WfMS über ihre komplette Lebendauer hinweg koordiniert und überwacht. Dabei sorgt das WfMS dafür, dass jeweils nur solche Aktivitäten einer WF-Instanz bearbeitbar sind, die der Ablauflogik zufolge aktuell zur Ausführung anstehen. Diese werden in die Arbeitslisten berechtigter Benutzer eingefügt. Welche Personen zur Bearbeitung einer bestimmten Aktivität autorisiert sind, wird dabei meist durch Angabe einer Rolle festgelegt, die auch den entsprechenden Bearbeitern zugeordnet sein muss. Dabei ist es möglich, dass mehrere Benutzer ermittelt werden, die für die Bearbeitung einer bestimmten Aktivität in Frage kommen.

Generell besteht bei (zentralen) WfMS das Problem, dass der einzelne WF-Server durch die vielen zu erledigenden Aufgaben (z.B. Aktualisieren von Benutzerarbeitslisten, interpretative Ausführung von WF-Graphen, Starten von Aktivitätenprogrammen) und durch die hohe Anzahl der zu kontrollierenden WF-Instanzen überlastet sein kann. Zur Vermeidung von Überlastsituationen in WfMS wurden in der Vergangenheit zahlreiche Ansätze für ein verteiltes WF-Management entwickelt (z.B. [Bau01, MWW+98, SNS99]). Ihnen liegt allesamt die Idee zugrunde, dass die zu bewältigende Last auf mehrere WF-Server aufgeteilt wird. Des Weiteren wird üblicherweise versucht, für die Kontrolle der Aktivitäten jeweils einen gut geeigneten WF-Server auszuwählen. Hierfür bietet sich z.B. der WF-Server desjenigen Teilnetzes an, in welchem die Mehrzahl der potenziellen Bearbeiter der jeweiligen Aktivität angesiedelt ist. Durch die topologische Nähe der entsprechenden WF-Clients zum betroffenen WF-Server ergibt sich bei dieser Art der Serverzuordnung ein günstiges Kommunikationsverhalten (vgl. [Bau01, BD99]), d.h., es resultieren kurze Antwortzeiten und niedrige Kommunikationskosten.

Üblicherweise wird bei den in der Literatur diskutierten Ansätzen für verteiltes WF-Management der für eine bestimmte WF-Aktivität zuständige WF-Server bereits bei der Modellierung des betreffenden Workflow festgelegt. In bestimmten Situationen kann es jedoch sinnvoll sein, von diesen vorgeplanten Serverzuordnungen abzuweichen. Eine auch aus anderen Bereichen der Informatik bekannte Vorgehensweise ist in diesem Zusammenhang, die Zuordnung von Aufgaben zu Hardwarekomponenten erst zur Ausführungszeit durchzuführen. Ein Beispiel hierfür ist das Prozess-Scheduling in Betriebssystemen (vgl. [CK88, Gos91]). Hier wird der Prozessor für die Ausführung eines Betriebssystemprozesses meist erst bei dessen Start ausgewählt. Auch bei der verteilten Systemumgebung CORBA [OMG95] sowie gängigen Applikationsservern wird der Server zur Bearbeitung eines Methodenaufrufs ggf. erst zur Laufzeit ausgewählt. Dabei kann die aktuelle Last- und Ausfallsituation berücksichtigt werden.

Wird diese weit verbreitete Vorgehensweise auf verteilte WfMS übertragen, kann dadurch etwa versucht werden, die Überlastung einer WfMS-Komponente (WF-Server, Teilnetz oder Gateway) zu kompensieren. Ist zum Beispiel ein bestimmter WF-Server überlastet, können Aufgaben, für die er ursprünglich vorgesehen war, einem anderen, augenblicklich weniger belasteten Server übertragen werden. Ein anderes Szenario, in dem von einer dynamischen (also zur Laufzeit der WF-Instanzen stattfinden-

Diese Arbeit wurde im Rahmen des von der Deutschen Forschungsgemeinschaft (DFG) geförderten Projektes »Skalierbarkeit in adaptiven Workflow-Management-Systemen« erstellt.

den) Veränderung der Serverzuordnungen – im Folgenden kurz *dynamische Serveränderungen* genannt – Gebrauch gemacht werden kann, ist der Ausfall von Komponenten des WfMS. In diesem Fall kann (dynamisch) ein anderer WF-Server für die Kontrolle der betroffenen Aktivitäten bzw. Aktivitäteninstanzen gewählt werden.

Aus den dargelegten Gründen erscheint die Verwendung dynamischer Serveränderungen – zumindest bei oberflächlicher Betrachtung – sehr vorteilhaft zu sein. Wir haben deshalb systematisch untersucht, ob und wie dynamische Serveränderungen zur Lösung der genannten Probleme (Überlastung bzw. Ausfall von Systemkomponenten) sinnvoll verwendet werden können. Interessanterweise hat sich dabei herausgestellt, dass dynamische Serveränderungen für die Behandlung von Ausfall- und Überlastsituationen nur bedingt geeignet sind. Der Hauptbeitrag dieser Arbeit besteht darin, aufzuzeigen, warum dem so ist und welche speziellen Schwierigkeiten mit einem solchen Ansatz verbunden sind. Des Weiteren wird dargelegt, wie der Überlastung und dem Ausfall von WfMS-Komponenten durch alternative Verfahren, wie sie das von uns entwickelte verteilte WF-Ausführungsmodell ADEPT$_{distribution}$ (vgl. [Bau01, BD97, BRD01b]) bietet, geeigneter begegnet werden kann.

Der vorliegende Beitrag unterscheidet sich von unseren bisherigen Veröffentlichungen zu verteiltem Workflow-Management dadurch, dass die WF-Server im ADEPT$_{distribution}$-Ausführungsmodell den Aktivitäten bereits zur Modellierungszeit zugeordnet werden oder diese Zuordnung zumindest vorgeplant wird (vgl. [BD00]). Ziel dabei ist es primär, die Belastung des Kommunikationssystems zu reduzieren. In diesem Beitrag wollen wir untersuchen, welches Potenzial in einer dynamischen Veränderung dieser Serverzuordnungen bei der Ausführung der WF-Instanzen liegt.

Im nachfolgenden Abschnitt werden wichtige Grundlagen für verteiltes WF-Management skizziert. Sie sind für das weitere Verständnis dieses Beitrags notwendig. Abschnitt 3 untersucht, in welchen Zuständen einer WF-Instanz dynamische Änderungen von Serverzuordnungen möglich und sinnvoll sind. Der nachfolgende Abschnitt 4 widmet sich der Behandlung von Überlastsituationen,

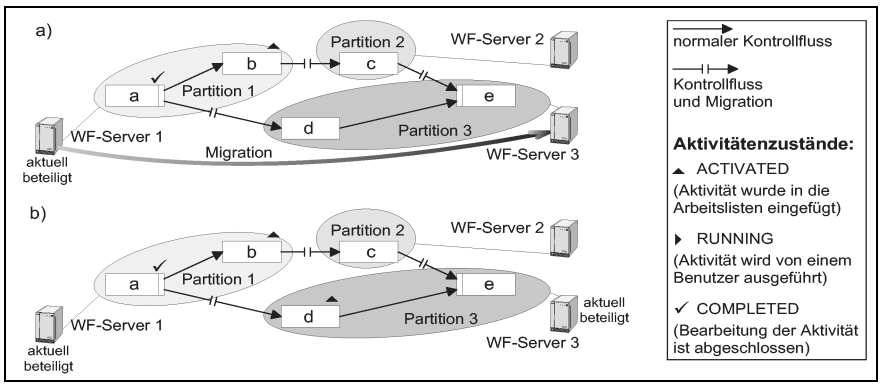

Abb. 1: a) Migration einer WF-Instanz und b) daraus resultierender Zustand der WF-Instanz

während Abschnitt 5 auf den Ausfall von Komponenten des verteilten WfMS eingeht. Abschnitt 6 behandelt Alternativen zu dynamischen Serveränderungen, mit denen sich die beschriebenen Problemstellungen besser lösen lassen. Abschnitt 7 diskutiert verwandte Ansätze. Der Beitrag schließt mit einer Zusammenfassung der gewonnenen Erkenntnisse.

2 Verteiltes Workflow-Management

Wir betrachten im Folgenden unternehmensweite oder -übergreifende WF-Anwendungen (vgl. [DR99, WWW+96]) im operativen Einsatz. In [KAGM96, SK97] etwa werden Szenarien beschrieben, bei denen die Zahl der Benutzer des WfMS bis auf einige zehntausend anwachsen kann oder mehrere zehntausend WF-Instanzen gleichzeitig im System ablaufen können. Durch diese hohe Anzahl von Benutzern und gleichzeitig ausgeführten WF-Instanzen wird eine sehr große Last für die WF-Server und das Kommunikationssystem erzeugt. Als Folge können Komponenten des WfMS überlastet werden.

Zur Vermeidung von Überlastsituationen wird von verschiedener Seite vorgeschlagen, eine WF-Instanz nicht nur von einem einzigen WF-Server kontrollieren zu lassen, sondern ihren WF-Graphen geeignet zu partitionieren und abschnittsweise von verschiedenen WF-Servern zu steuern. Dieser Ansatz wird auch von ADEPT$_{distribution}$ (vgl. [Bau01, BD97]), der verteilten Variante des von uns entwickelten ADEPT-WfMS (vgl. [DRK00, RD98, RBFD01]), verfolgt (vgl. Abb. 1).

Wird nun bei der Ausführung einer WF-Instanz eines bestimmten WF-Typs das Ende einer Partition erreicht, wird ihre Kontrolle an den nächsten WF-Server weitergereicht. Um solche *Migrationen* zu ermöglichen, muss jeweils eine Beschreibung des Zustands der WF-Instanz (aktueller Ausführungszustand und aktuelle Werte von Parameterdaten der Aktivitätenprogramme) zu diesem Server transportiert werden (vgl. [BRD01b]).

Wie bereits erwähnt, verwenden auch andere Ansätze eine solche Partitionierung von WF-Graphen zur Modellierzeit und die zugehörigen Migrationen während der Ausführungszeit (z.B. [CGP+96, MWW+98]). Einige dieser Ansätze verfolgen zusätzlich das Ziel, die Kommunikationskosten zu minimieren. Konkrete Erfahrungen mit existierenden WfMS haben gezeigt, dass zwischen einem WF-Server und seinen WF-Clients sehr viel kommuniziert wird und zum Teil auch große Datenmengen ausgetauscht werden (vgl. [Mar01]). Dies kann dazu führen, dass das Kommunikationssystem überlastet wird. Deshalb wird in ADEPT$_{distribution}$ zur Modellierzeit für jeden Aktivitätentyp automatisch derjenige WF-Server berechnet, durch dessen Verwendung der Gesamtkommunikationsaufwand minimiert wird (vgl. [BD97, BD00]). In der Regel wird dabei der WF-Server einer Aktivität so gewählt, dass er im Teilnetz der Mehrzahl ihrer potenziellen Bearbeiter liegt. Allerdings wird bei der Berechnung der Serverzuordnung auch berücksichtigt, dass die Migrationen selbst ebenfalls (Netzwerk-)Last verursachen (vgl. [BD97]).

ADEPT$_{distribution}$ erfüllt darüber hinaus die Konsistenzbedingung, dass die verteilte Ausführung von Workflows äquivalent (bzgl. der den Benutzern ermöglichten Aktionen) zu einer von einem zentralen WF-Server gesteuerten WF-Ausführung ist (vgl. [Bau01, WW97]). Dabei

ist die Systemarchitektur von $ADEPT_{distribution}$ so gestaltet, dass die verschiedenen WF-Server auf getrennten (jeweils lokal platzierten) Datenbanken arbeiten. Die Synchronisation und der Austausch von Daten zwischen diesen WF-Servern (und damit auch zwischen den Datenbanken) erfolgt ausschließlich durch die schon erwähnten Migrationen. (Die Anwendungsdaten werden, soweit dies möglich ist, ebenfalls vom WfMS verwaltet, so dass auch diese in Migrationen einbezogen werden können.)

Die beschriebene Vorgehensweise ist wesentlich effizienter (vgl. hierzu [BD97, BD99, BRD01b])[1] als etwa eine Replikation der Daten auf Datenbankebene, da hier zusätzlich Anwendungswissen genutzt werden kann. Aus demselben Grund ist das WfMS besser in der Lage, (durch eine geeignete Serverauswahl) die Kommunikationslast zu reduzieren, als dies durch eine tiefer liegende Schicht (z.B. das Betriebssystem) möglich wäre. Die WF-Clients sind mit mehreren WF-Servern verbunden, ggf. sogar mit allen WF-Servern des WfMS. Allerdings befindet sich normalerweise genau ein WF-Server in ihrer Domäne. Zu diesem besteht, wegen dessen lokaler Lage, eine besonders schnelle (und auch kostengünstige) Verbindung. Deshalb ist es vorteilhaft, wenn die von einem WF-Client ausgeführten Aktivitäteninstanzen vom lokal gelegenen WF-Server kontrolliert werden.

3 Realisierungsvarianten für dyn. Serveränderungen

Die Art und Weise, wie dynamische Serveränderungen durchgeführt werden, hat großen Einfluss auf die entstehenden Kosten und den resultierenden Nutzen. In diesem Abschnitt untersuchen wir die generellen Möglichkeiten zur Durchführung von Migrationen, bei denen der Zielserver erst zur Ausführungszeit festgelegt wird (*dynamische Migration*). Die dynamischen Migrationen sollen dazu dienen, ein für die Benutzer des WfMS günstiges Verhalten zu erzielen (*Quality of Service*).

1. In [BRD01b] wird ein Verfahren vorgestellt, mit dem nicht der gesamte Datenkontext der migrierten WF-Instanz übertragen werden muss, sondern nur die tatsächlich für die weitere WF-Ausführung noch benötigten Daten.

Es geht also darum, die Verfügbarkeit und das Antwortzeitverhalten des WfMS zu verbessern. Eine Erhöhung des WfMS-Durchsatzes ist nicht unser Ziel, da dies für die Benutzer des WfMS keinen unmittelbaren Vorteil bringt. (Eine Erhöhung des Durchsatzes lässt sich im Falle von Überlastungen am einfachsten durch die Suspendierung von WF-Instanzen erreichen, allerdings auf Kosten der Antwortzeiten. Der für die Durchführung von Migrationen notwendige Aufwand kann den Gesamt-Systemdurchsatz natürlich nur verschlechtern.)

Damit die Reaktion auf die Überlastung oder den Ausfall von WfMS-Komponenten möglichst zeitnah erfolgen kann, sollte der WF-Server der aktuell zu bearbeitenden Aktivität(en) modifiziert werden. Um eine größere Wirkung zu erzielen und um unnötige Migrationen zu verhindern, empfiehlt es sich dabei, stets die Serverzuordnung der gesamten aktuellen Partition zu verändern. Entscheidend ist hierbei der Zeitpunkt, zu dem dies geschieht. Prinzipiell bestehen die folgenden Möglichkeiten (vgl. Abb. 2):

Ansatz 1:
Dynamische Serveränderungen können zu beliebigen Zeitpunkten erfolgen
Bei diesem Ansatz werden dynamische Abweichungen von den zur Modellierungszeit festgelegten Serverzuordnungen zu beliebigen Zeitpunkten zugelassen. Dies hat natürlich zur Folge, dass der ursprünglich vorgesehene WF-Server alle in die Bearbeitung der aktuellen Aktivität involvierten WF-Clients über die Änderung informieren muss. Dies wird notwendig, da Rückmeldungen zur aktuellen Aktivität von den betreffenden WF-Clients ansonsten an den falschen (nämlich veralteten) WF-Server gerichtet werden. Außerdem wird mit diesem Ansatz bei Serveränderungen meist eine zusätzliche Migration notwendig.

Ansatz 2:
Dynamische Serveränderungen sind für bereits aktivierte, aber noch nicht selektierte Aktivitäten unzulässig
Bei dieser Vorgehensweise erfolgen keine dynamischen Migrationsentscheidungen, wenn sich die Aktivität (des betroffenen Ausführungszweigs) aktuell im Zustand ACTIVATED befindet. In diesem Zustand wartet das WfMS darauf, dass einer der (evtl. zahlreichen) potenziellen Bearbeiter der Aktivität diese aus seiner Arbeitsliste auswählt und startet (vgl. [Bau01, RD98]). Der Ausschluss dynamischer Migrationen für diesen Fall hat den Vorteil, dass dann noch maximal ein Client (nämlich derjenige, der die Aktivität tatsächlich bearbeitet) über die Serveränderung informiert werden muss.

Ansatz 3:
Dynamische Serveränderungen sind nur nach Beendigung von Aktivitäten erlaubt
Sind dynamische Migrationen nur dann möglich, wenn aktuell ein Übergang zwischen zwei aufeinander folgenden Aktivitäten stattfindet, dann ist augenblicklich auch kein WF-Client in den betrachteten Ausführungszweig der WF-Instanz involviert. (Die abgeschlossene Aktivität steht in keiner Arbeitsliste mehr, wird also nicht mehr bearbeitet, und für die nachfolgende Aktivität gibt es noch keine Arbeitslisteneinträge.) Deshalb muss auch kein WF-Client über die dynamische Serveränderung informiert werden; WF-Clients müssen bei Ansatz 3 nicht einmal wissen, dass solche Serveränderungen zur Laufzeit möglich sind. Es muss lediglich eine Migration durchgeführt werden, um die dynamische Serveränderung umzusetzen.

Ansatz 4:
Dynamische Serveränderungen sind nur bei ohnehin vorgesehenen Migrationen bzw. Partitionswechseln erlaubt
Werden dynamische Migrationsentscheidungen nur gefällt, wenn an der entsprechenden Stelle des WF-Graphen zur Modellierungszeit ohnehin eine Migration vorgesehen worden ist, dann ist lediglich der Zielserver der (anstehenden) Migration zu verändern. Bei dieser Vorgehensweise werden insbesondere keine zusätzlichen Migrationen notwendig.

Wir wollen nun untersuchen, wie effizient die vier vorgestellten Varianten sind: Die Ansätze 1 und 2 sind in ihrer Realisierung wesentlich aufwendiger als die beiden anderen Ansätze, weil in die dynamische Serveränderung evtl. WF-Clients einbezogen werden müssen. Außerdem erfordern sie mehr Kommunikation und sind fehleranfälliger. Fehler können z.B. entstehen, wenn ein WF-Client, der vorübergehend nicht erreichbar war, nun die Ausgabedaten eines beendeten Aktivitä-

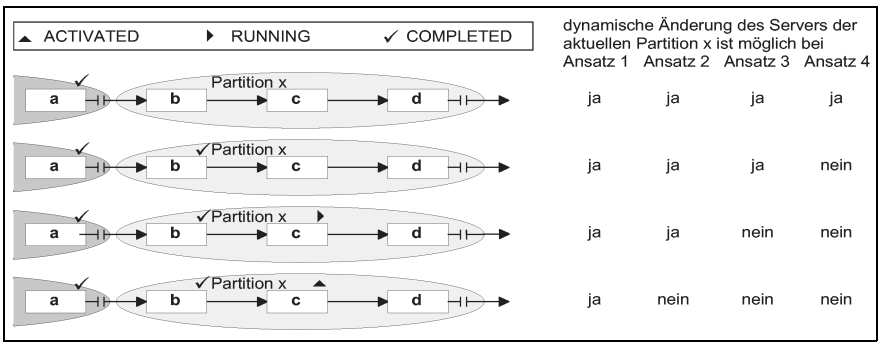

Abb. 2: Realisierungsvarianten für dynamische Serveränderungen

tenprogramms an einen inzwischen veralteten WF-Server zurückgeben will.

Der Ansatz 3 erzeugt ebenfalls Kommunikationslast, da bei dynamischen Serveränderungen zusätzliche Migrationen durchgeführt werden müssen. Diese bringen stets einen gewissen Aufwand mit sich, insbesondere für das Kommunikationssystem (vgl. [BD99]).

Nur der Ansatz 4 erfordert keine zusätzliche Kommunikation und hat auch keine Auswirkungen auf die WF-Clients. Da keine Migrationen notwendig werden, entspricht dieser Ansatz auch am ehesten den Schedulingverfahren für Betriebssystemprozesse (vgl. [Gos91]). Bei diesen wird normalerweise angenommen, dass (noch) kein Prozesskontext existiert oder dass dieser ohnehin auf allen beteiligten Knoten vorhanden ist. Deshalb müssen bei einem Prozessorwechsel (fast) keine Daten übertragen werden. Hierzu vergleichbar ist, dass bei einer dynamischen Serverwahl kein Aufwand für eine zusätzliche Migration anfällt.

Dynamische Migrationen sind aber beim Ansatz 4 (aufgrund des geforderten Bearbeitungszustands) für weniger WF-Instanzen möglich. Der Grund dafür ist, dass hier nur solche Instanzen migriert werden können, bei denen gerade eine (vorgeplante) Migration erfolgt. Dennoch dürften sich bei diesem Ansatz weiterhin genügend WF-Instanzen für dynamische Migrationen eignen, da ein WF-Server in dem eingangs beschriebenen Szenario zu jedem Zeitpunkt sehr viele WF-Instanzen kontrolliert. Dies gilt insbesondere für die betrachteten Hochleistungs-WfMS. Generell ist ohnehin nicht zu empfehlen, dynamische Serveränderungen für zu viele WF-Instanzen auf einmal durchzuführen, da dies zur Unterlastung des vormals überlasteten Servers führen könnte, also zu einem »Schwingen« der Last.

Um uns nicht unnötig einzuschränken, werden im Folgenden, trotz der erwähnten Schwächen bzw. Nachteile der Ansätze 1 bis 3, jeweils alle Ansätze weiterverfolgt. Dabei wird untersucht, welche Ansätze zur Lösung der jeweiligen Problemstellung am besten geeignet sind.

4 Behandlung von Überlastsituationen

Im Folgenden wird untersucht, inwieweit die Ansätze 1 bis 4 geeignet sind, um eine Überlastung bestimmter Komponenten des WfMS zu kompensieren. Dabei wird zuerst auf Überlastungen einzelner Komponenten des Kommunikationssystems (Teilnetze, Gateways) und dann auf Überlastungen von WF-Servern eingegangen. Im Anschluss daran werden Fragestellungen diskutiert, die das Verhalten von dynamischen Serveränderungen im Allgemeinen betreffen.

4.1 Überlastung von Komponenten des Kommunikationssystems

Überlastung von Teilnetzen

Es wird das in Abbildung 3 dargestellte Szenario unterstellt: Ein Teilnetz *x* des WfMS ist augenblicklich überlastet, so dass inakzeptable Antwortzeiten entstehen. Diese Überlastung soll behoben werden, indem WF-Instanzen, die von dem WF-Server dieses Teilnetzes kontrolliert werden sollen, einem anderen WF-Server zugeordnet werden.

Leider ist es durch dynamische Serveränderungen nicht möglich, das Teilnetz *x* zu entlasten, wenn der Großteil der potenziellen Bearbeiter der anstehenden Aktivitäten diesem Teilnetz angehört. Die Kommunikation zwischen den WF-Clients dieser Benutzer und dem WF-Server wird in einem solchen Fall weiterhin durch das Teilnetz *x* laufen, da die Clients in diesem Teilnetz angesiedelt sind. Deshalb wird das überlastete Teilnetz nicht signifikant entlastet.

Ungünstigerweise bildet die beschriebene Bearbeiterverteilung aber den Regelfall. Meist befindet sich ein Großteil der potenziellen Bearbeiter im Teilnetz desjenigen WF-Servers, welcher die aktuelle Partition kontrolliert. Dieser Server wurde – sinnvollerweise – gerade wegen dieser Bearbeiterverteilung ausgewählt, mit dem Ziel Kommunikationskosten einzusparen (vgl. Abschnitt 2). Dieses Ziel wird durch Einsatz dynamischer Migrationen jedoch konterkariert.

Wird für die dynamische Serveränderung einer der Ansätze 1 bis 3 aus Abschnitt 3 verwendet, so wird die Gesamtsituation durch die zusätzlich notwendig werdende Migration zunächst sogar noch verschlechtert. Der Grund dafür ist, dass eine Migration mit dem WF-Server des Teilnetzes *x* als Quellserver immer auch zu einer Belastung von Teilnetz *x* führt. Eine Verwendung der Ansätze 1 oder 2 verschlechtert die Lastsituation weiter, da hier die dynamische Serveränderung auch noch zusätzliche Kommunikationen mit Clients erforderlich macht.

Überlastung von Gateways

Ist ein Gateway (z.B. eine Fernkommunikationsleitung) überlastet, so kann eine ähnliche Vorgehensweise wie bei überlasteten Teilnetzen gewählt werden: WF-Instanzen werden dynamisch zu einem WF-Server migriert, dessen Teilnetz nicht an dieses Gateway angeschlossen ist. Allerdings ergeben sich bei dieser Vorgehensweise ähnliche Probleme wie im vorherigen Abschnitt beschrieben, so dass sie als nicht empfehlenswert zu betrachten ist. Die erzielten Vorteile sind sogar geringer, da mit dem Teilnetz des ursprünglich vorgesehenen WF-Servers mehrere Gateways verbunden sind, die durch die dynamische Migration alle entlastet werden. Die Entlastung des Gateways fällt dadurch nur anteilsmäßig aus.

4.2 Überlastung von WF-Servern

Von einem überlasteten WF-Server sprechen wir, wenn dieser die aktuell anstehenden Aufgaben nicht mehr (vollständig) bewältigen kann. Ein solcher WF-Server kann entlastet werden, indem WF-Instanzen von ihm »wegmigriert« werden bzw. WF-Instanzen, für deren Bearbei-

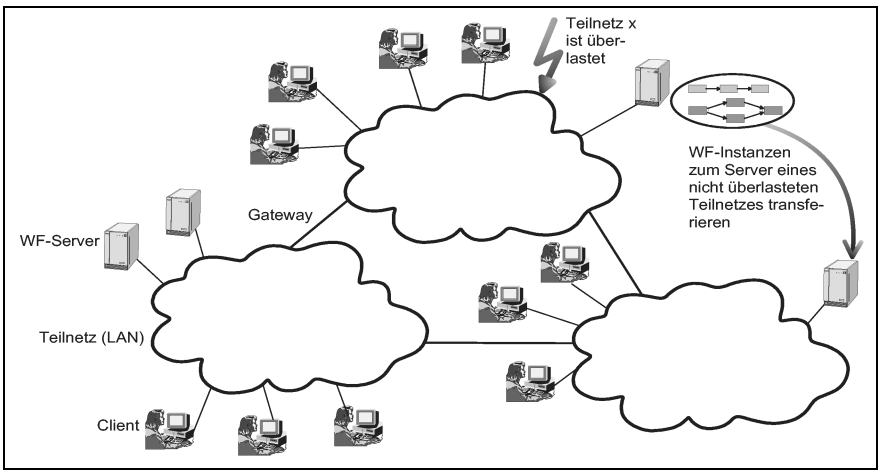

Abb. 3: Dynamische Änderung von Serverzuordnungen bei Überlastung eines Teilnetzes

tung er vorgesehen war, erst gar nicht zu ihm transferiert werden.

Bei diesem Szenario führen dynamische Serveränderungen schon eher zum gewünschten Ziel, da der WF-Server tatsächlich weniger WF-Instanzen kontrollieren muss. Allerdings wird ein überlasteter WF-Server bei den Ansätzen 1 bis 3 (vgl. Abschnitt 3) durch die zusätzliche Migration weiter belastet. Da der Aufwand für die zusätzliche Migration (wegen der größeren Datenmenge) häufig wesentlich größer ist als für eine reguläre Aktivitätenausführung (siehe hierzu [BRD01b]), kann sich das Verfahren sogar als nachteilig erweisen. Bei den Ansätzen 1 und 2 kommen für den WF-Server bei der dynamischen Serveränderung auch noch Interaktionen mit den aktuellen Clients hinzu, was zusätzlichen Aufwand bedeutet. Außerdem verschlechtern die im nachfolgenden Abschnitt beschriebenen generellen Nachteile dynamischer Migrationen bei der Behebung von Überlastsituationen die Güte des diskutierten Ansatzes weiter.

4.3 Allgemeine Betrachtungen zu dynamischen Serveränderungen in Überlastsituationen

Nachdem wir die Verfahren für dynamische Migrationen bei Überlastung von Komponenten des Kommunikationssystems bzw. bei Überlastung von WF-Servern skizziert haben, wenden wir uns nun generellen Schwierigkeiten der Verfahren zu.

Wie alle Verfahren zur Lastbalancierung erfordern auch die vorgestellten Ansätze einen Austausch von Lastinformation (in irgendeiner Form). Dabei ist irrelevant, welches Verfahren konkret verwendet wird. Vom verteilten Scheduling von Betriebssystemprozessen sind hierzu zahlreiche Varianten bekannt (vgl. [Gos91]). So kann z.B. periodisch ein Lastinformationsaustausch zwischen den WF-Servern stattfinden oder bei einer anstehenden Migrationsentscheidung kann bei potenziellen Zielservern nachgefragt werden, wie stark diese aktuell belastet sind. Bei all diesen Varianten trägt die Verwendung von dynamischen Serveränderungen selbst zur Überlastung von Komponenten des WfMS bei. Allerdings kann dieser Nachteil durch die Verwendung eines »Huckepack-Verfahrens« (vgl. [Tan96]) abgeschwächt werden: Die (relativ kleine) auszutauschende Lastinformation wird bei anderen, ohnehin notwendigen Kommunikationen mitübertragen. Falls angenommen werden kann, dass zwischen allen Paaren von WF-Servern ausreichend häufig solche Kommunikationen (z.B. Migrationen) stattfinden, dann werden für den Lastinformationsaustausch keine zusätzlichen Nachrichten notwendig, sondern es wird lediglich das zu übertragende Datenvolumen (geringfügig) erhöht.

Ein gravierenderer Nachteil lastabhängiger dynamischer Serveränderungen resultiert aus den in der Praxis häufig auftretenden langen »Pausen« bei der Abarbeitung einzelner WF-Instanzen. Diese Pausen können einerseits aus eventuellen Prioritäten der Bearbeiter resultieren, andererseits gibt es Workflows, bei denen ein aktivierter Schritt für Tage oder gar Wochen ruht, bevor er tatsächlich ausgeführt wird. (Dieser Fall kann z.B. im medizinischen Bereich bei lang andauernden Therapieprozessen auftreten.) Eine zur Entlastung einer Komponente dynamisch migrierte WF-Instanz kann also längere Zeit ruhen. Folglich wird in diesem Fall kurzfristig überhaupt keine Entlastung erreicht. Die Entlastung der Komponente findet erst viel später statt, zu einem Zeitpunkt, an dem sie möglicherweise sogar unterlastet ist. Bei Verwendung der Ansätze 1 bis 3 ist die Situation noch schlechter: Die Migration, die zur Entlastung durchgeführt wird (und bei den Ansätzen 1 und 2 auch die notwendig werdende Interaktion mit den Clients), wird sofort ausgeführt, weshalb zusätzlicher Aufwand zu einem Zeitpunkt entsteht, an dem die Überlastung noch andauert. Dieser negative Effekt ist bei WfMS wesentlich stärker ausgeprägt als beim Betriebssystem-Scheduling, da WF-Instanzen eine wesentlich größere Datenmenge umfassen als übliche Betriebssystemprozesse. Außerdem werden beim Betriebssystem-Scheduling üblicherweise ausschließlich noch nicht gestartete Prozesse transferiert, so dass kein Prozesskontext zu übertragen ist.

4.4 Fazit

In Anbetracht der diskutierten Nachteile sind dynamische Serveränderungen wenig geeignet, um der Überlastung von Komponenten des Kommunikationssystems begegnen zu können. Lediglich eine Entlastung von WF-Servern ist theoretisch möglich, doch auch diese tritt erst deutlich verzögert ein. Wird ein entsprechendes Verfahren gewählt, so sollte unbedingt der Ansatz 4 aus Abschnitt 3 verwendet werden, um den WF-Server nicht zusätzlich zu belasten. Dynamische Migrationen haben sich in Überlastsituationen also als eher ungeeignet erwiesen.

5 Behandlung von Ausfallsituationen

Um auf den Ausfall von WF-Servern, Teilnetzen und Gateways reagieren zu können, wird folgende Idee umgesetzt: (Partitionen von) WF-Instanzen, die von dem Ausfall betroffen und deshalb blockiert sind, werden dynamisch einem anderen WF-Server zugeordnet. Wie wir noch sehen werden, spielt es dabei keine Rolle, ob dadurch der Ausfall eines WF-Servers oder einer Komponente des Kommunikationssystems kompensiert werden soll. Deshalb werden diese Fälle im Folgenden gemeinsam betrachtet.

5.1 Eigenschaften des Verfahrens

Soll auf den Ausfall einer WfMS-Komponente durch dynamische Veränderung von Serverzuordnungen reagiert werden, ist es nicht möglich, die betroffenen WF-Instanzen von ihrem aktuellen Server »wegzumigrieren«. Das heißt, die Ansätze 1 bis 3 scheiden aus. Eine solche Migration würde die Funktionsfähigkeit der ausgefallenen Komponente voraussetzen. Für WF-Server und Teilnetze ist diese Aussage unmittelbar ersichtlich, da diese Komponenten in einer entsprechenden Migration stets direkt involviert sind. Der Ausfall eines Gateways ist nur dann relevant, wenn dies zu einer Partitionierung des Netzwerkes führt. Andernfalls kann jeder WF-Server jeden WF-Client weiterhin erreichen. Im Falle einer Netzpartitionierung wäre aber eine dynamische Migration in den anderen Teil des Netzwerkes notwendig, damit der WF-Server die dort angesiedelten WF-Clients anschließend erreichen kann. Eine solche Migration ist aber aufgrund des Ausfalls der Kommunikationsverbindung ausgeschlossen. Aus den erläuterten Gründen bleibt im Zusammenhang mit der Kompensation von Komponentenausfällen lediglich die Verwendung von Ansatz 4. Für Benutzer, die aufgrund der Blockierung einer WF-Instanz auf die Bearbeitung ihrer Aktionen warten müssen, ist aber mit einem solchen Verfahren keine Verbesserung zu erzielen. Sie können weiterhin bestimmte, in ihrer Arbeitsliste angebotene Aktivitäten nicht starten bzw. von ihnen aktuell bearbeitete Aktivitäten nicht beenden, da der zugehörige WF-Server nicht erreichbar ist. Diese, für den Anwender sichtbaren Probleme können durch dynamische Serveränderungen nicht behoben werden.

Die Verwendung von Ansatz 4 führt in Ausfallsituationen lediglich dazu, dass für eine noch anstehende Migration einer WF-Instanz ein anderer Zielserver gewählt wird. Dies ist aber nur von sehr beschränktem Nutzen, weil aktuell keine Anwender auf die Ausführung einer zu dieser WF-Instanz gehörenden Aktion warten (die entsprechende WF-Instanz befindet sich nämlich in einer Migration, also unmittelbar vor der Aktivierung der nachfolgenden Aktivität). Die im vorherigen Absatz beschriebenen, gravierenden Probleme treten hier also ohnehin nicht auf. Da außerdem anzunehmen ist, dass eine ausgefallene Komponente zeitnah repariert wird, resultiert aus der entstehenden, nach außen kaum sichtbaren Verzögerung kein Problem, das den für das vorgestellte Verfahren erforderlichen Aufwand rechtfertigen würde.

5.2 Fazit

Wie die vorgestellte Analyse ergeben hat, ist die dynamische Änderung von Serverzuordnungen ein wenig geeignetes Verfahren, um den Ausfall von Komponenten des WfMS zu kompensieren. Hierfür sind Verfahren wie die Verwendung von Backup-Servern (vgl. [KAGM96]) wesentlich besser geeignet. Auf entsprechende Ansätze wird, ebenso wie auf Verfahren zur Behandlung der Überlastungsproblematik, kurz in Abschnitt 6 eingegangen.

Wegen der mangelnden Tauglichkeit dynamischer Serveränderungen erübrigt es sich hier, die skizzierten Verfahren formaler vorzustellen oder mögliche Realisierungsvarianten miteinander zu vergleichen (z.B. mögliche Alternativen bei der Auswahl des Zielservers einer dynamischen Migration, d.h. bei der Beantwortung der Fragestellung, wie eine dynamisch festgelegte Serverzuordnung konkret aussieht). Aus demselben Grund macht es ebenfalls wenig Sinn, auf Verfahren zum (effizienten) Erkennen von Komponentenausfällen näher einzugehen. (Dasselbe gilt für Verfahren zur effizienten Realisierung des Lastinformationsaustauschs und für Kriterien, ab wann und wie lange auf eine Überlastung reagiert werden sollte.) Schließlich erübrigt es sich, die Leistungsfähigkeit der entsprechenden Verfahren durch eine Modellrechnung oder Simulation zu evaluieren, da ihre Mängel offensichtlich sind.

6 Alternative Vorgehensweisen

Da sich dynamische Serveränderungen als wenig geeignet erwiesen haben, um die Überlastungs- und Ausfallproblematik in (verteilten) WfMS zu lösen, wenden wir uns im Folgenden kurz einigen alternativen Verfahren zu.

Die Fragestellung, wie sich Überlastungen des Kommunikationssystems vermeiden lassen, ist schon immer ein zentraler Aspekt von ADEPT$_{distribution}$ gewesen. Dazu haben wir verschiedene Verfahren entwickelt, mit denen die in einem (verteilten) WfMS zu kommunizierende Datenmenge erheblich reduziert werden kann. Diese Verfahren können auch in Kombination miteinander verwendet werden.

Das erste Verfahren (vgl. [BD97]) unterstützt den WF-Modellierer bei der Bestimmung *optimaler Serverzuordnungen* für WF-Aktivitäten. Ziel dabei ist die Minimierung der bei der späteren WF-Ausführung entstehenden Gesamtkommunikationskosten. Um dies zu erreichen, wurden von uns Algorithmen entwickelt, die ausgehend von dem Prozess- und Organisationsmodell und unter Verwendung eines Kostenmodells die bei der WF-Ausführung entstehenden Kommunikationskosten abschätzen. Unter Nutzung dieser Information werden dann (möglichst) optimale Serverzuordnungen automatisch berechnet.

Der zweite Ansatzpunkt betrifft *variable Serverzuordnungen* (vgl. [BD00]), bei deren Verwendung sich Datenübertragungen immer dann einsparen lassen (vgl. [BD99]), wenn die potenziellen Bearbeiter einer Aktivität von Vorgängeraktivitäten abhängen (z.B. eine Aktivität soll vom selben Benutzer bearbeitet werden wie eine bestimmte Vorgängeraktivität). In diesen Fällen werden flexible Serverzuordnungen verwendet, bei denen der tatsächlich ausgewählte WF-Server ebenfalls von Vorgängeraktivitäten abhängig ist, sich also erst zur Ausführungszeit der WF-Instanz ergibt.

Drittens wurden von uns Verfahren entwickelt, mit denen die bei Migrationen zu übertragende Datenmenge deutlich reduziert werden kann (vgl. [BRD01b]). So wird verhindert, dass Zustandsinformation einer WF-Instanz oder Parameterdaten von Aktivitätenprogrammen durch wiederholte Migrationen zum selben WF-Server mehrfach (redundant) an diesen übertragen werden (z.B. in Verbindung mit Schleifen).

Sollten all diese Maßnahmen immer noch nicht ausreichen, um eine Überlastung des Kommunikationssystems zu verhindern, müssen kleinere Teilnetze gebildet werden. Da dann von jedem Teilnetz eine kleinere Anzahl von Benutzern bedient werden muss, kann dessen Überlastung verhindert werden.

Auch die Problematik einer Überlastung von WF-Servern wurde im ADEPT-Projekt betrachtet. In [Bau01] wird ein

Verfahren vorgestellt, durch das eine solche Überlastung ausgeschlossen werden kann. Dieses ermöglicht die Verwendung von beliebig vielen WF-Servern in demselben Teilnetz (Domain). Außerdem kann die zu bewältigende Last in einem beliebigen (vorzugebenden) Verhältnis auf diese Server verteilt werden. Da das Verfahren während der Steuerung von WF-Instanzen keine zusätzliche Kommunikation generiert, trägt es auch nicht zur Überlastung des Kommunikationssystems bei.

Dem Ausfall von Komponenten des Kommunikationssystems oder von WF-Servern begegnet man am günstigsten durch den redundanten Einsatz von Hardware, falls eine sehr hohe Verfügbarkeit gefordert ist. So werden in [KAGM96] Verfahren vorgestellt, die den Einsatz eines Backup-Servers erlauben. Die zur Steuerung der WF-Instanzen notwendige Information wird dann auch an diesen Backup-Server ständig übermittelt, so dass er im Falle eines Serverausfalls die Kontrolle übernehmen kann. Der Ausfall von Komponenten des Kommunikationssystems wird am besten durch redundante Kommunikationshardware kompensiert (z.B. durch eine doppelte Ringbildung wie bei FDDI [Tane96]).

7 Diskussion

Verfahren, die auf der Verwendung von Lastinformation basieren, werden beim Scheduling von Betriebssystemprozessen seit langem mit großem Erfolg eingesetzt (vgl. [CK88, Gos91]). Bei diesen Verfahren wird (meist aufgrund der aktuellen Lastsituation) festgelegt, welchem Prozessor ein bestimmter Betriebssystemprozess zugeteilt werden soll. Allerdings wird der entsprechende Prozessor normalerweise schon beim Start eines Betriebssystemprozesses bestimmt (*non-preemptive Scheduling* [CK88]), so dass kein laufender Prozess transferiert werden muss. (Dies würde einer Migration entsprechen.)

Zu beachten ist in diesem Zusammenhang, dass in Bezug auf unsere Fragestellung die Betriebssystemprozesse nicht den Prozessinstanzen des WfMS entsprechen. Eine WF-Instanz besteht aus Aktivitäteninstanzen, die den WF-Servern zugeordnet werden sollen, ebenso wie eine Anwendung aus Betriebssystemprozessen besteht, die den Prozessoren zugeteilt werden. Betriebssystemprozesse entsprechen also den Aktivitäteninstanzen. Bei der Zuordnung von Aktivitäteninstanzen zu WF-Servern besteht der Vorteil, dass nützliche Metainformation zu den Aktivitäten zur Verfügung steht (z.B. die Bearbeiterzuordnung oder die Verteilung der Benutzer auf die Teilnetze), was bei Betriebssystemprozessen normalerweise nicht der Fall ist. Deshalb ist es beim Scheduling von WF-Instanzen möglich, einen bzgl. des resultierenden Kommunikationsverhaltens sehr viel günstigeren WF-Server auszuwählen, als dies durch reine Lastbalancierung möglich wäre. Die Nutzung solcher Informationen entspricht der Verwendung von statischen Schedulingverfahren (vgl. [CK88]) in Betriebssystemen anstelle der dort üblichen dynamischen Verfahren.

Wir möchten an dieser Stelle auf eine ausführliche Darstellung von Ansätzen für verteiltes WF-Management verzichten, da sich eine solche z.B. in [Bau01, BD99] findet. Die meisten Ansätze legen, ebenso wie $\text{ADEPT}_{distribution}$, die Zuordnung der Aktivitäteninstanzen zu den WF-Servern auf Basis des Prozess- und des Organisationsmodells fest. So wird in MENTOR [MWW+98] und WIDE [CGP+96] der WF-Server nahe bei den potenziellen Bearbeitern der aktuellen Aktivität allokiert. Dasselbe gilt für MOBILE [HS96], wobei allerdings eine WF-Instanz während ihrer gesamten Lebensdauer vom selben WF-Server kontrolliert wird. Deshalb sind keine Migrationen notwendig. Allerdings können bei diesem Ansatz Subprozesse von einem anderen WF-Server kontrolliert werden (vgl. [SNS99]). Dieser wird zur Laufzeit aufgrund verschiedener Kriterien (z.B. Rechte, Gewichte) ausgewählt. Bei METEOR2 [DKM+97], CodAlf [SM96] und BPA-Frame [SM96] wird der WF-Server stets nahe bei der zur aktuellen Aktivität gehörenden Anwendung gewählt. Schließlich finden sich in der Literatur auch noch voll verteilte Ansätzen (z.B. Exotica/FMQM [AMG+95] und INCAS [BMR96]), bei denen jeweils der Rechner des aktuellen Benutzers die Serverfunktionalität übernimmt. All diese Ansätze führen das Scheduling der Aktivitäteninstanzen aber unabhängig von der aktuellen Last- und Ausfallsituation durch.

Dass es in WfMS auch möglich ist, das Scheduling unabhängig von den Prozess- und Organisationsmodellen zu realisieren, zeigt der Exotica/Cluster-Ansatz (vgl. [AKA+94]). Bei diesem wird der WF-Server (Cluster)[1] für eine WF-Instanz bei deren Start zufällig ausgewählt. In diesem Cluster verbleibt sie für ihre gesamte Lebenszeit, es werden also keine bereits gestarteten WF-Instanzen zu einem anderen Cluster migriert. Allerdings verwendet auch dieser Ansatz für die Auswahl des WF-Servers keine Lastinformation. (Er kann aber leicht dahingehend modifiziert werden.) Außerdem wurde in [Bau01, BD99] gezeigt, dass sich wegen der fehlenden Nutzung von Modellinformation bei diesem Ansatz ein sehr ungünstiges Kommunikationsverhalten ergibt.

In der WF-Literatur werden auch CORBA-basierte Ansätze für verteiltes WF-Management vorgestellt: WASA_2 [Wes99] etwa realisiert eine verteilte WF-Ausführung, indem die Aktivitäten als CORBA-Objekte betrachtet werden, die sich Zustandsänderungen signalisieren. Da diese Objekte auf beliebigen Rechnern platziert werden können, lässt sich so ein verteiltes Workflow-Management realisieren. Im MOKASSIN-Projekt wird die Verteilung (vgl. [JH99]) ähnlich wie beim vorherigen Ansatz realisiert: Die Aktivitäteninstanzen repräsentieren CORBA-Objekte, die durch Ereignisse miteinander kommunizieren und die auf beliebigen Rechnern platziert werden können. Bei diesen Ansätzen kann durch die CORBA-Middleware prinzipiell jede beliebige Verteilung realisiert werden. Dabei wäre es auch möglich, dass die Verteilung so vorgenommen wird, dass Last- und Ausfallinformation berücksichtigt wird. Allerdings würden bei einer solchen Vorgehensweise die in dieser Arbeit diskutierten Schwierigkeiten auftreten.

8 Zusammenfassung

In diesem Beitrag haben wir untersucht, inwieweit durch eine dynamische Veränderung von Serverzuordnungen auf Überlastsituationen und Komponentenausfälle

1. Genau genommen besteht bei dem in [AKA+94] beschriebenen Ansatz ein Cluster aus mehreren WF-Servern mit gemeinsamer WF-Datenbank. Diese Server sind alle fähig, die WF-Instanzen des Clusters zu steuern. Da zwischen diesen Rechnern aber keine Migrationen (in unserem Sinne) stattfinden, wollen wir sie als einen einzigen WF-Server betrachten.

in einem verteilten WfMS reagiert werden kann. Dabei hat sich gezeigt, dass die Verfahren bei Überlastungen wenig geeignet sind, da sie für Teilnetze und Gateways häufig fast gar keine Entlastung ermöglichen. Der für das jeweilige Verfahren notwendige Zusatzaufwand kann sogar dazu führen, dass sich die Gesamtsituation noch verschlechtert. Lediglich eine Entlastung von WF-Servern ist – allerdings sehr eingeschränkt – möglich. Auch zur Kompensation von Ausfällen einzelner WfMS-Komponenten sind dynamische Serveränderungen wenig geeignet. Diejenigen Auswirkungen der Ausfälle, die für den Endanwender wirklich kritisch sind, können durch ihre Verwendung nämlich nicht vermindert werden, weil der Ausfall der Komponente die Durchführung der dynamischen Migration verhindert. Insgesamt betrachtet ist deshalb die Verwendung von dynamischen Serveränderungen nicht zu empfehlen.

In diesem Beitrag wurden auch noch Alternativen zu dynamischen Serveränderungen skizziert. Allerdings war es dabei nicht das Ziel, diese im Detail zu behandeln (dies erfolgte in anderen Publikationen), sondern es sollten nur mögliche Alternativen zur Verwendung von dynamischen Serveränderungen aufgezeigt werden. Mit diesen Verfahren ist es möglich, sowohl Überlastsituationen von Netzwerkkomponenten und WF-Servern als auch Ausfällen dieser WfMS-Komponenten geeignet zu begegnen.

Zusammenfassend lässt sich also feststellen, dass dynamische Migrationen wenig geeignet sind, um die gegebenen Problemstellungen zu lösen. Da es Alternativen zu diesem Ansatz gibt, mit denen die gewünschten Ziele durchaus erreicht werden können, besteht auch keine Notwendigkeit, diese Verfahren einzusetzen.

Danksagung: Wir danken den anonymen Gutachtern für ihre hilfreichen Anregungen.

Literatur

[AH02]] *van der Aalst, W. M. P.; van Hee, K.*: Workflow Management. MIT Press, 2002.

[AKA+94] *Alonso, G.; Kamath, M.; Agrawal, D.; El Abbadi, A.; Günthör, R.; Mohan, C.:* Failure Handling in Large Scale Workflow Management Systems. Technischer Bericht RJ9913, IBM Almaden Research Center, November 1994.

[AMG+95] *Alonso, G.; Mohan, C.; Günthör, R.; Agrawal, D.; El Abbadi, A.; Kamath, M.:* Exotica/FMQM: A Persistent Message-Based Architecture for Distributed Workflow Management. In: Proc. IFIP Working Conf. on Information Systems for Decentralized Organisations, Trondheim, August 1995.

[Bau01] *Bauer, T.:* Effiziente Realisierung unternehmensweiter Workflow-Management-Systeme. Dissertation, Universität Ulm, Fakultät für Informatik, Februar 2001 (erschienen beim Tenea-Verlag Berlin).

[BD97] *Bauer, T.; Dadam, P.:* A Distributed Execution Environment for Large-Scale Workflow Management Systems with Subnets and Server Migration. In: Proc. 2nd Conf. on Cooperative Information Systems, S. 99–108, Kiawah Island, SC, Juni 1997.

[BD99] *Bauer, T.; Dadam, P.*: Verteilungsmodelle für Workflow-Management-Systeme – Klassifikation und Simulation. Informatik Forschung und Entwicklung, 14(4):203–217, Dezember 1999.

[BD00] *Bauer, T.; Dadam, P.*: Efficient Distributed Workflow Management Based on Variable Server Assignments. In: Proc. 12th Conf. on Advanced Information Systems Engineering, S. 94–109, Stockholm, Juni 2000.

[BMR96] *Barbará, D.; Mehrotra, S.; Rusinkiewicz, M.:* INCAs: Managing Dynamic Workflows in Distributed Environments. Journal of Database Management, Special Issue on Multidatabases, 7(1):5–15, 1996.

[BRD01a] *Bauer, T.; Reichert, M.; Dadam, P.:* Adaptives und verteiltes Workflow-Management. In: Proc. Datenbanksysteme in Büro, Technik und Wissenschaft, S. 47–66, Oldenburg, März 2001.

[BRD01b] *Bauer, T.; Reichert, M.; Dadam, P.:* Effiziente Übertragung von Prozessinstanzdaten in verteilten Workflow-Management-Systemen. Informatik Forschung und Entwicklung, 16(2):76–92, Juni 2001.

[CGP+96] *Casati, F.; Grefen, P.; Pernici, B.; Pozzi, G.; Sánchez, G.*: WIDE: Workflow Model and Architecture. CTIT Technical Report 96-19, University of Twente, 1996.

[CK88] *Casavant, T. L.; Kuhl, J. G.:* A Taxonomy of Scheduling in General-Purpose Distributed Computing Systems. IEEE ToSE, 14(2):141–154, Februar 1988.

[DKM+97] *Das, S.; Kochut, K.; Miller, J.; Sheth, A.; Worah, D.*: ORBWork: A Reliable Distributed CORBA-based Workflow Enactment System for METEOR2. Technical Report #UGACS-TR-97-001, Dept. of Comp. Science, University of Georgia, Februar 1997.

[DR99] *Dadam, P.; Reichert, M.*: Proc. Workshop on Enterprise-Wide and Cross-Enterprise Workflow-Management – Concepts, Systems, Applications. 29. Jahrestagung der GI (Informatik'99), Paderborn, Oktober 1999.

[DRK00] *Dadam, P.; Reichert, M.; Kuhn, K.:* Clinical Workflows – The Killer Application for Process-oriented Information Systems? In: Proc. 4th Int. Conf. on Business Information Systems, S. 36–59, Posen, April 2000.

[Gos91] *Goscinski, A.*: Distributed Operating Systems: The Logical Design. Addison-Wesley, 1991.

[HS96] *Heinl, P.; Schuster, H.:* Towards a Highly Scaleable Architecture for Workflow Management Systems. In: Proc. 7th Int. Workshop on Database and Expert Systems Applications, S. 439–444, Zürich, September 1996.

[JBS97] *Jablonski, S.; Böhm, M.; Schulze, W.:* Workflow-Management: Entwicklung von Anwendungen und Systemen; Facetten einer neuen Technologie. dpunkt.verlag, 1997.

[JH99] *Joeris, G.; Herzog, O.:* Towards Flexible and High-Level Modeling and Enacting of Processes. In: Proc. 11th Int. Conf. on Advanced Information Systems Engineering, S. 88–102, Heidelberg, Juni 1999.

[KAGM96] *Kamath, M.; Alonso, G.; Günthör, R.; Mohan, C.*: Providing High Availability in Very Large Workflow Management Systems. In: Proc. 5th Int. Conf. on Extending Database Technology, S. 427–442, Avignon, März 1996.

[LR00] *Leymann, F.; Roller, D.:* Production Workflow – Concepts and Techniques. Prentice Hall, 2000.

[Mar01] *Martschat, U.:* Vergleich und Bewertung von Production-Workflow-Management-Systemen. Diplomarbeit, Universität Ulm, Fakultät für Informatik, 2001.

[MWW+98] *Muth, P.; Wodtke, D.; Weißenfels, J.; Kotz-Dittrich, A.; Weikum, G.:* From Centralized Workflow Specification to Distributed Workflow Execution. Journal of Intelligent Information Systems, Special Issue on Workflow Management Systems, 10(2):159–184, März/April 1998.

[Obe96] *Oberweis, A.:* Modellierung und Ausführung von Workflows mit Petri-Netzen. Teubner-Verlag, 1996.

[OMG95] *OMG:* The Common Object Request Broker: Architecture and Specification. Technischer Bericht Revision 2.0, Object Management Group, Juli 1995.

[RBFD01] *Reichert, M.; Bauer, T.; Fries, T.; Dadam, P.:* Realisierung flexibler, unternehmensweiter Workflow-Anwendungen mit ADEPT. In: Proc. Conf. Elektronische Geschäftsprozesse, S. 217–228, Klagenfurt, September 2001.

[RD98] *Reichert, M.; Dadam, P.:* $\text{ADEPT}_{\text{flex}}$ – Supporting Dynamic Changes of Workflows Without Losing Control. Journal of Intelligent Information Systems, Special Issue on Workflow Management Systems, 10(2):93–129, März/April 1998.

[SK97] *Sheth, A.; Kochut, K. J.:* Workflow Applications to Research Agenda: Scalable and Dynamic Work Coordination and Collaboration Systems. In: Proc. NATO Advanced Study Institute on Workflow Management Sys-

tems and Interoperability, S. 12–21, Istanbul, August 1997.

[SM96] *Schill, A.; Mittasch, C.:* Workflow Management Systems on Top of OSF DCE and OMG CORBA. Distributed Systems Engineering, 3(4):250–262, Dezember 1996.

[SNS99] *Schuster, H.; Neeb, J.; Schamburger, R.:* A Configuration Management Approach for Large Workflow Management Systems. In: Proc. Int. Joint Conf. on Work Activities Coordination and Collaboration, S. 177–186, San Francisco, Februar 1999.

[Tan96] *Tanenbaum, A. S.:* Computer Networks. Prentice Hall, 1996.

[Wes99] *Weske. M.:* Workflow Management Through Distributed and Persistent CORBA Workflow Objects. In: Proc. 11th Int. Conf. on Advanced Information Systems Engineering, Heidelberg, 1999.

[WW97] *Wodtke, D.; Weikum, G.*: A Formal Foundation for Distributed Workflow Execution Based on State Charts. Proc. Int. Conf. on Database Theory. Delphi, Januar 1997.

[WWW96+] *Wodtke, D.; Weißenfels, J.; Weikum, G.; Kotz-Dittrich, A.*: The Mentor Project – Steps Towards Enterprise-Wide Workflow Management. Proc. 12th Int'l Conf. on Data Engineering, S. 556-565, New Orleans, März 1996.

Dr. Thomas Bauer studierte Informatik in Ulm, wo er 2001 auch promovierte. Der Titel seiner Dissertation lautet »Effiziente Realisierung unternehmensweiter Workflow-Management-Systeme«. Seit Januar 2002 arbeitet Dr. Bauer im DaimlerChrysler Forschungszentrum in Ulm im Bereich *Engineering Process Management.*

Dr. Thomas Bauer
DaimlerChrysler Forschung
und Technologie
Abteilung RIC/ED
Postfach 2360
89013 Ulm
Thomas.TB.Bauer@DaimlerChrysler.com
http://www.daimlerchrysler.com

Dr. Manfred Reichert ist wissenschaftlicher Assistent in der Abteilung Datenbanken und Informationssysteme der Universität Ulm. Hier promovierte er im Juli 2000 zum Thema »Dynamische Ablaufänderungen in Workflow-Management-Systemen«. Aktuelle Arbeitsgebiete sind unternehmensweite und -übergreifende WF-Anwendungen sowie verschiedene Aspekte von Workflow-Management-Systemen.

Dr. Manfred Reichert
Universität Ulm
Fakultät für Informatik
Abt. Datenbanken und Informationssysteme
James-Franck-Ring
89069 Ulm
reichert@informatik.uni-ulm.de
http://www.informatik.uni-ulm.de/dbis/