ulm university universität ulm

Vergleich und Bewertung physischer und logischer Fragmentierungsansätze im Umfeld mobiler Prozesse

Bachelorarbeit

an der Universität Ulm
Fakultät für Ingenieurwissenschaften und Informatik
Institut für Datenbanken und Informationssysteme

vorgelegt von:

Jörg Grüning

November 2010

Gutachter: Prof. Dr. Manfred Reichert

Betreuer: Dipl. Inf. Rüdiger Pryss

Vergleich und Bewertung physischer und logischer Fragmentierungsansätze im Umfeld mobiler Prozesse

Jörg Grüning

Universität Ulm, Institut DBIS
joerg.gruening@uni-ulm.de

Zusammenfassung Flexibilität, um auf sich schnell verändernde Situationen reagieren zu können, ist ein immer wichtigerer Aspekt im Prozessumfeld. Zur Erreichung dieser Flexibilität ist die mobile Nutzung von Prozessen, die sinnvoll nur durch die Fragmentierung von Prozessen möglich ist, von großer Bedeutung. Herkömmliche Workflow- beziehungsweise Prozessmanagementsysteme [35] mit zentralen Workflow-/Prozess-Management-Servern sind für Einsätze in verteilten mobilen Umgebungen durch die starke Fokussierung auf lokale Ressourcen zumeist wenig geeignet. Der zentrale Aspekt dieser Arbeit ist die Darstellung verschiedener theoretischer Ansätze aus dem Umfeld der verteilten Prozesse zur Unterstützung mobiler Prozesse. Aus den derzeit existierenden, theoretischen Ansätzen, die auch bereits teilweise umgesetzt wurden, sind aus Sicht dieser Arbeit, die beiden wichtigen Konzepte der physischen Fragmentierung sowie der logischen Fragmentierung, bei der Prozessinstanzen migriert werden, interessant, andere Ansätze, wie zum Beispiel agentenbasierte Ansätze, werden in dieser Arbeit nicht betrachtet. Abschließend wird am Beispiel eines kommerziellen Produktes - dem Microsoft BizTalk Server - gezeigt, inwiefern dieses zur Unterstützung mobiler Prozesse eingesetzt werden kann.

1 Einführung

In der heutigen, von großem Wettbewerb geprägten globalisierten Welt, ist es für eine profitorientierte Organisation unerlässlich, die internen Prozesse sowie die Schnittstellen zu kooperierenden Unternehmen zu kennen. Um angemessen auf sich ständig ändernde Bedingungen im eigenen Umfeld reagieren zu können, müssen die Prozesse konstant überwacht und optimiert werden. Die tendenziell immer stärker werdende Forderung, zu nahezu jeder Zeit und von jedem Ort aus auf Daten und Dienste Zugriff zu haben, stellt hohe Anforderungen an die an der Bereitstellung beteiligten Unternehmen. Um der Forderung nach Flexibilität [30,31] Rechnung zu tragen, ist es notwendig, Prozesse in großen Teilen durch den Einsatz mobiler Techniken zu unterstützen.

Aufgrund der zahlreichen heterogenen Eigenschaften mobiler Umgebungen und der vielfältigen Einsatzmöglichkeiten, sind herkömmliche, zentral ge-

steuerte, Workflow-/Prozessmanagementsysteme für die Kontrolle der mobilen Prozesse aus *technischer* oder *organisatorischer* Sicht häufig ungeeignet oder zu *starr* [1]. So können zum Beispiel aufgrund sicherheitsrelevanter Hindernisse Dienste nicht angesprochen werden oder technische Schwierigkeiten wie Verbindungsabbrüche erschweren einen reibungslosen Ablauf des mobilen Prozesses. Daher liegt der Fokus dieser Arbeit auf der Darstellung und Untersuchung verschiedener Ansätze auf dem Gebiet der verteilten Prozessausführung und deren Eignung für mobile Prozesse.

2 Typische Einsatzszenarien

Folgende exemplarische Einsatzszenarien sind für den Einsatz mobiler Unterstützung denkbar:

1: Aufnehmen von Bestellungen durch eine Servicekraft in einem Restaurant, die dann im weiteren Prozess zur Verfügung stehen (Insofern die Bestellung als Prozess abläuft). So können diese unter anderem direkt nach der Aufnahme der Bestellung an die Küche weitergeleitet werden, um die georderten Produkte zeitnah zuzubereiten. Ein weiterer Punkt in diesem Szenario ist die Möglichkeit aus den Bestelldaten, ohne ein weiteres Eintippen der Rechnungsdaten in eine Kasse, direkt bei Zahlungswunsch eines Kunden, mit dem Abkassieren zu beginnen.

2: Abbildung 1 zeigt einen Beispielprozess im Umfeld von Außendienstmitarbeitern, die beispielsweise im Lebensmittelgroßhandel tätig sind. Diese können beim Besuch einer zugeordneten Discounterfiliale, die Bestandskontrolle der Waren ihrer Firma mobil durchführen und aufgrund der erfassten Bestandsdaten abgleichen, wieviel von einer Ware geordert werden muss. So kann die Bestellung der Fehlmenge zeitnah an ein Logistikzentrum des Großhändlers gesendet werden, ohne diese beispielsweise auf Papierformularen erfasste Bestellung erst per Fax vom Home-Office zu versenden.

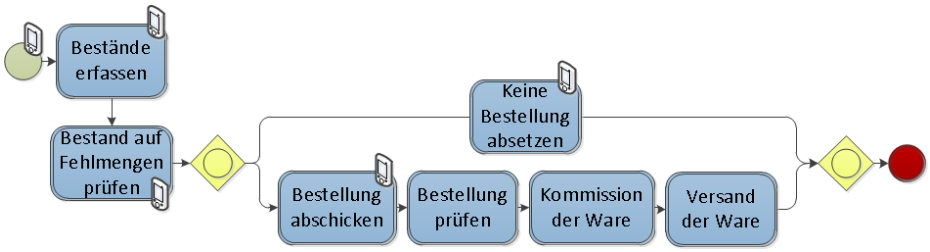

Abbildung 1: Beispielprozess Außendienst in BPMN-Darstellung

3: Bei der Visite der Patienten durch Ärzte in Kliniken ist es erforderlich, die Daten der Patienten vor Ort verfügbar zu haben, um den Verlauf der Behandlung, verordnete Medikamente oder andere Informationen nachschlagen

zu können. Durch den Einsatz mobiler Unterstützung ist es möglich, auf diese Daten, ohne sie vorab aus stationären PCs auszulesen oder als herkömmliche Patientenakten mitzunehmen, zuzugreifen. Des Weiteren ist auch das Eintragen neuer Befunde und Bemerkungen oder die Änderung der Medikamentierung des Patienten möglich.

Im folgenden Abschnitt werden die Struktur und der Aufbau der Arbeit dargestellt. Neben einer kurzen Zusammenfassung der Inhalte jedes Kapitels werden die Kerninhalte der Arbeit zusätzlich graphisch abgebildet.

3 Struktur und Aufbau der Arbeit

Kapitel 4 beschäftigt sich mit den grundlegenden Merkmalen von Prozessen sowie mobilen Prozessen. Hierzu werden die Merkmale mobiler Devices sowie die Herausforderungen im Umfeld mobiler Prozesse herausgearbeitet. Der Kernpunkt dieses Kapitels ist die Definition mobiler Prozesse, wie diese im Rahmen dieser Arbeit verstanden werden.

In Kapitel 5 werden die Anforderungen mobiler Prozesse an Workflow / Prozess-Management-Systeme dargestellt. Nach der Darstellung dieser Anforderungen werden die beiden verschiedenen Ansätze der verteilten Ausführung, die für diese Arbeit relevanten Ansätze, der physischen Fragmentierung und der Migration von Prozessen aufgezeigt, was den zentralen Teil der Arbeit darstellt. Abschließend wird ein Zwischenfazit über die Eignung der verschiedenen betrachteten Ansätze für einen Einsatz im Umfeld mobiler Prozesse gezogen. In Abbildung 2 wird der Aufbau der Arbeit dargestellt und die Inhalte der Kernkapitel der Arbeit graphisch veranschaulicht (Kapitel 4 + 5).

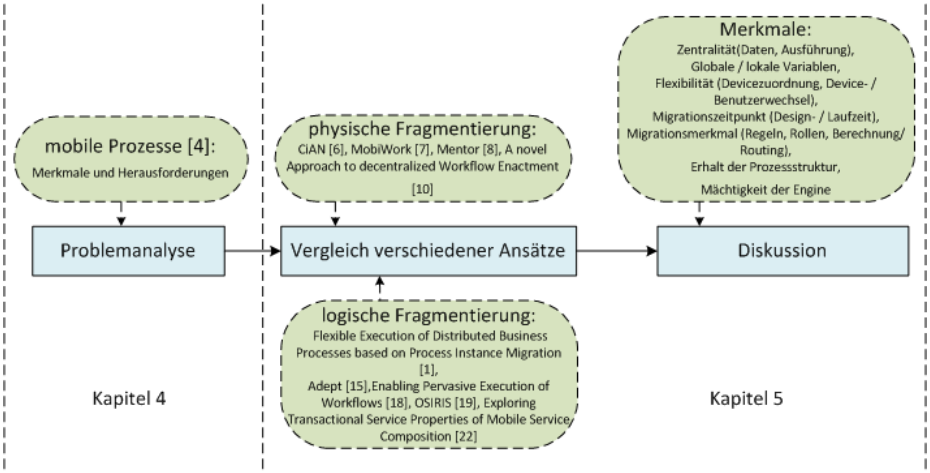

Abbildung 2: Übersicht der Arbeit

Kapitel 6 stellt mit dem Microsoft BizTalk Server exemplarisch ein kommerzielles System vor, dass unter anderen auch Business-Process-Management Funktionalität [12] zur Verfügung stellt. Nach einem kurzen Anriss der Architektur des Systems und dessen Hauptkomponenten, wird dieses System auf seine Eignung für den Einsatz im Umfeld der mobilen Prozessunterstützung untersucht. In Kapitel 7 schließt die Arbeit mit einem Fazit und gibt einen Ausblick.

4 Prozesse

Unter einem Prozess (Abbildung 2) [3] versteht man eine Menge von Aktivitäten, die in einem meist deterministischen Ablauf, nacheinander beziehungsweise parallel von beteiligten Akteuren ausgeführt werden. Zweck der Ausführung dieser Aktivitäten ist das Erreichen eines gemeinsamen Ziels, wie beispielsweise der Umwandlung von verschiedensten Inputs in Outputs, wie einem Produkt oder einer Dienstleistung eines Industrieunternehmens [13]. Zur Steuerung dieser Prozesse kommen Workflow-/Prozess-Management-Systeme zum Einsatz, welche das Deployen und Instanziieren der Workflows / Prozesse, den Austausch von Daten, wie zum Beispiel den von den Prozessteilnehmern erzeugten Prozessdaten, den Umgebungsdaten der Prozesse oder den Prozess-Meta-Daten, die den Prozess beschreiben, übernehmen. Des Weiteren legen Workflow-/Prozess-Management-Systeme die operationale Semantik fest und übernehmen das Monitoring der Prozesse.

Abbildung 3: Beispielprozess in BPMN-Darstellung

Der Begriff der Verteilung [21] steht in engem Zusammenhang mit Prozessen [2]. Da in der Regel nicht alle Aktivitäten eines Prozesses auf einem einzelnen System ausgeführt werden können oder das beispielsweise aus sicherheitsrelevanten Gründen nicht gewünscht ist, müssen diese auf verschiedene Systeme verteilt werden. Durch die Verteilung von Prozessen auf mehrere Devices beziehungsweise Server wird es möglich, die Performanz einer gesamten Prozessausführung zu erhöhen, indem beispielsweise Teile des Prozesses an externe Dienstleister ausgelagert werden oder Aktivitäten oder ganze Teilprozesse parallel ausgeführt werden [3]. Um diese Verteilung von Prozessen erst zu ermöglichen, ist es nötig Prozesse zu fragmentieren. Abbildung 4 zeigt die Choreographie eines BPMN-Prozesses, die jedoch zur Fragmentierung von Prozessen genutzt werden kann. Ein Fragment eines Prozesses stellt eine Teilmenge aller Aktivitäten eines Prozesses dar, die in engem Kontext zueinander stehen. Die genaue Bestimmung einer solchen Teilmenge ist nicht festgelegt und wird unterschiedlich gehandhabt.

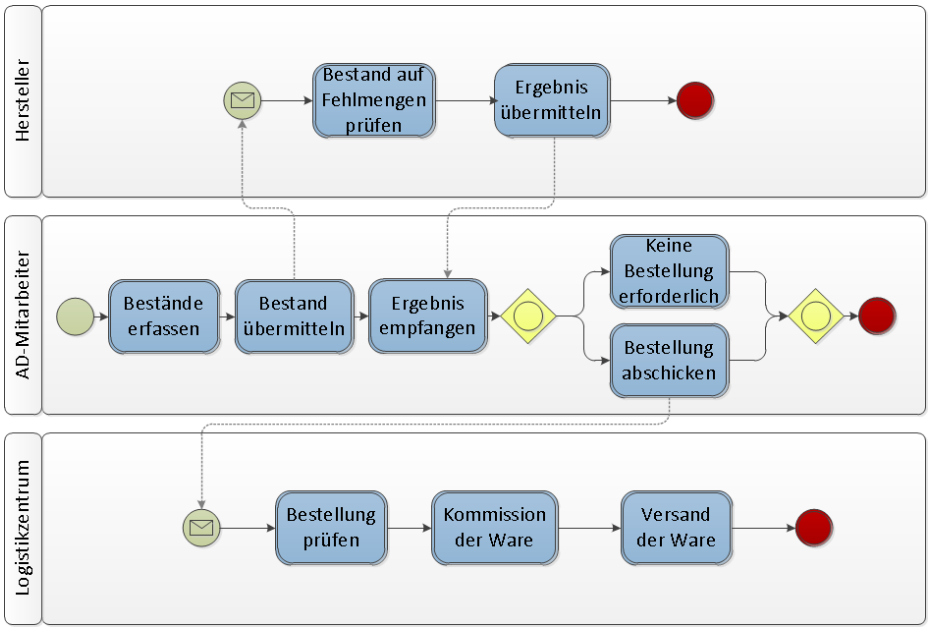

Abbildung 4: Choreographie eines Prozess in BPMN-Darstellung

Sind an der Ausführung der Aktivitäten mobile Devices, wie beispielsweise PDA, PocketPC's oder Ähnliches beteiligt, spricht man von einem mobilen Prozess. Im folgenden Abschnitt werden die Merkmale mobiler Prozesse sowie deren Umgebung aufgezeigt.

4.1 Mobile Prozesse

Wie bereits im letzten Abschnitt angedeutet, handelt es sich um einen mobilen Prozess (Abbildung 5), sobald Fragmente, d.h. einzelne oder mehrere Aktivitäten eines Prozesses mobil ausgeführt werden sollen [4]. Die oben erwähnte Ausführung einer Aktivität auf einem mobilen Device, wie PDA, PocketPcs oder auch Notebooks führt zu Herausforderungen bei mobilen Aktivitäten, die im Folgenden betrachtet werden.

Mobile Aktivitäten sind nicht ausschließlich durch die Ausführung auf einem mobilen Gerät definiert. So stellt die Ausführung von Tätigkeiten, die von einem Mitarbeiter kurzzeitig an einem anderen Ort innerhalb der Firma auf einem Notebook ausgeführt werden, keinen mobilen Prozess dar [4], obwohl der Einsatz eines mobilen Devices wie eines Notebook dies suggerieren mag. Für die Definition einer mobilen Aktivität sind weitere Bedingungen, wie die extern bestimmte Unsicherheit des Ortes und die Kooperation mit externen Ressourcen notwendig. Unter der extern bestimmten Unsicherheit des Ortes versteht man, dass der Ort, an dem mobile Aktivitäten zur Ausführung kommen, in verschiedenen Instanziierungen des mobilen Prozesses auch verschieden ist,

und, dass dieser durch den Akteur, der die Aktivität ausführt, nicht frei bestimmt werden kann. Erfüllt eine Aktivität die oben genannten Bedingungen so spricht man von einer mobilen Aktivität, wie man sie in mobilen Prozessen ausführt. Zur besseren Übersicht, sind in Tabelle 1 die Merkmale mobiler Aktivitäten abgebildet.

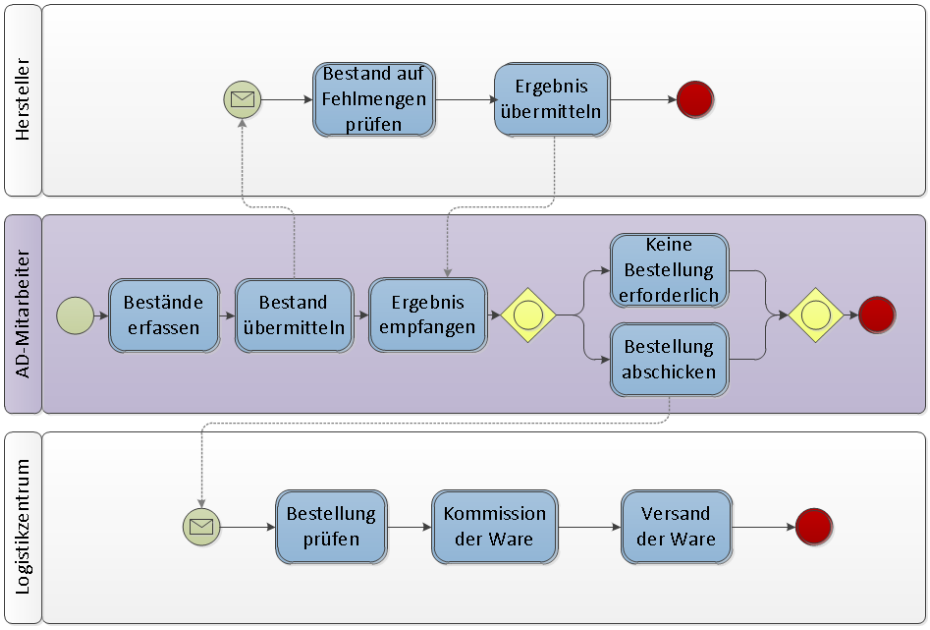

Abbildung 5: Möglicher mobiler Teilprozess der Choreographie in BPMN-Darstellung

In der Literatur existiert eine Vielzahl von Definitionen mobiler Prozesse. Diese oft stark differierenden Interpretationen fassen mobile Prozesse unterschiedlich weit und schränken diese dadurch teilweise ein. Im folgenden werden daher exemplarisch zwei aus verschiedenen Definitionen resultierende Einschränkungen aufgezeigt.

In [4] definiert die Autorin einen mobilen Prozess als einen Prozess, der eine oder mehrere mobile Aktivitäten beinhaltet, was jedoch dadurch beschränkt ist, dass kein Prozess komplett mobil sein kann, da die Start- und Endaktivitäten des Prozesses durch sogenannte fixe Devices ausgeführt werden müssen. So führt die Autorin als Beispiel einen *Ein-Mann-Monteur-Betrieb* an, der seine Prozesse und benötigten Daten auf einem mobilen Device mit sich führe und mangels fehlender Kommunikation mit externen Systemen kein mobiler Prozess sei. Im Rahmen dieser Arbeit, wird ein solcher Definition mobiler Prozess von [4] als zu einschränkend betrachtet.

Merkmal	**Beschreibung**
Unsicherheit des Ortes	Je nach Prozessinstanz kann der Ort, an dem eine Aktivität ausgeführt wird, unterschiedlich sein. So kann es auch vokommen, dass sich der Ort während der Ausführung ändert.
externe Bestimmtheit des Ortes	Der Ort der Ausführung einer mobilen Aktivtät kann von externen Faktoren, wie zum Beispiel dem Ort, an dem ein Schaden entstanden ist, der nun durch einen Monteur vor Ort repariert werden soll, bestimmt sein und somit eine freie Wahl des Ortes durch den ausführenden Benutzer verhindern.
Interaktion mit externen Ressourcen	Bei der Ausführung mobiler Aktivitäten ist die Interaktion, mit aus Sicht des Prozesses externen Ressourcen, wie anderen Benutzern oder Objekten, erforderlich.

Tabelle 1: Herausforderungen mobiler Aktivitäten

Eine weitere Definition mobiler Prozesse wird in [40] gegeben. Hier wird ein mobiler Prozess als eine Folge von Aktivitäten, die über eine längere Zeit auf mehreren Devices ausgeführt werden und als Ergebnis, den gewünschten Effekt des Initiators des Prozesses haben. Die Arbeit führt als Ziel an, die Möglichkeiten mobiler Devices durch die Zusammenarbeit mehrerer Devices zu erhöhen, was erreicht werden soll indem verteilte Prozesse in passende mobile Systeminfrastrukturen integriert werden sollen. Daraus kann man schließen, dass die mobilen Prozesse stark an der verfügbaren Infrastruktur ausgerichtet werden und gegebenenfalls an diese angepasst werden. Auch die Orientierung an bestehender Infrastruktur ist im Kontext dieser Arbeit eine unerwünschte Einschränkung der Flexibilität mobiler Prozesse.

Im Rahmen dieser Arbeit versteht man unter einem mobilen Prozess daher einen Prozess, bei dem mindestens eine Aktivität mobil ist, das heißt, dass sie auf einem mobilen Device ausgeführt wird. Ein mobiler Prozess kann jedoch auch größere mobile Prozessfragmente beinhalten oder sogar vollständig mobil ausgeführt werden, d.h alle Aktivitäten beinhalten. Auch die Ausführung eines Prozesses von der Instanziierung, dem Start der Prozessausführung bis zum Beenden der Ausführung, auf einem einzelnen mobilen Device wird als mobiler Prozess verstanden. Das Design des mobilen Prozesses orientiert sich hierbei ausschließlich am Geschäftsmodell, der durch den mobilen Prozess abgebildet werden soll. Die mobile Infrastruktur wird nur als Mittel zur Ausführung der Prozesses verstanden und muss eine ausreichend große Flexibilität bieten, um vielfältige Geschäftsprozesse abzubilden ohne Einschränkungen zu verur-

sachen.

Im folgenden werden die Merkmale der in mobilen Prozessen eingesetzten mobilen Devices betrachtet und dabei auftretende Herausforderungen aufgezeigt.

4.2 Merkmale mobiler Devices

Durch die mobile Ausführung der Prozessschritte und der Ausführung auf Devices mit sehr beschränkten Ressourcen, ist beim Design und der Ausführung mobiler Prozesse eine hohe Aufmerksamkeit nötig. Nach der im vorhergegangenen Abschnitt erfolgten Definition von mobilen Prozessen sollen daher in diesem Abschnitt die Herausforderungen, die im Umfeld mobiler Prozess auftreten, dargestellt werden. Die folgenden Merkmale stellen, die für diese Arbeit als relevant betrachteten Merkmalen mobiler Devices dar. Eine allgemeine Betrachtung der Herausforderungen mobiler Prozesse ohne eine Betrachtung der Devices erfolgt im folgendem Abschnitt.

Hardwarebeschränkungen Viele der auf dem Markt verfügbaren Devices, wie PocketPcs, PDAs oder unter gewissen Umständen auch einsetzbare Mobiltelefone, verfügen nur über geringe Hardwareressourcen. So sind die verbauten Prozessoren meist recht leistungsarm und können daher nicht für komplexere, rechenintensive Anwendungen verwendet werden. Die Ausstattung der Devices mit Speicher, sei es Arbeitsspeicher oder der Platz für die persistente Speicherung von Anwendungsteilen, wie auch lokaler Datenbanken ist ebenfalls häufig sehr knapp bemessen, wodurch sich größere Datenmengen kaum auf den Devices halten lassen.

Konnektivität Im Gegensatz zu stationären Devices, die in der Regel über Kabel- oder Funkverbindungen mit hohen Bandbreiten und einem hohen Maß an Zuverlässigkeit in den Prozess integriert sind, müssen mobile Devices ohne solche Leitungen auskommen. Da mobile Devices an den verschiedensten Standorten zum Einsatz kommen, kann es nicht immer gewährleistet werden, dass Verbindungen permanent verfügbar sind. So kann es bereits aufgrund von zu geringen Signalstärken, Funklöchern oder einstreuenden Störsignalen in mobilen Datennetzen zu Ausfällen oder beträchtlichen Einschränkungen der im Allgemeinen bereits relativ geringen Bandbreite kommen.

Verfügbarkeit Die bereits im vorhergehenden Punkt genannten Probleme im Zusammenhang mit der Konnektivität können die Verfügbarkeit mobiler Devices empfindlich schmälern. Allerdings besteht bei mobilen Devices in erhöhten Maße die Gefahr, dass ein Device aufgrund einer Fehlfunktion oder eines vollständigen Defekts nicht nur temporär für den mobilen Prozess, sondern ganz für die weitere Prozessausführung ausfällt. Jedoch ist bei einem mobilen Prozess nicht nur das Device als Fehlerquelle möglich, sondern auch Einflüsse der Umgebung oder der Benutzer, die das Device bedienen sollen, können die

Verfügbarkeit des Device für den Prozess beeinträchtigen. Wenn der Benutzer auf dem Weg zum Ausführungsort der mobilen Aktivität beispielsweise durch Verkehrsprobleme oder Erkrankungen aufgehalten wird [4], kann die Prozessausführung problematisch werden. Hier sind noch eine Menge weiterer Einflussfaktoren denkbar.

Standort Auch der Standort ist ein entscheidender Faktor im Umfeld mobiler Prozessausführung. Während an einigen Standorten die Konnektivität der Devices beeinträchtigt sein kann, hat der Standort auch noch weitere maßgebliche Faktoren auf die Ausführbarkeit mobiler Prozesse. Nicht an jeden Standort können benötigte, externe Dienste verfügbar sein oder Sicherheitsbestimmungen lassen den Zugriff auf die benötigen Ressourcen nicht zu. Zudem ist wie oben definiert die Unsicherheit des Ortes eine problematische, nicht immer vorhersehbare Größe. Durch die externe Bestimmtheit des Ortes entstehen zahlreiche Faktoren, die nicht vorhersehbar sind, wenn zum Beispiel ein Techniker eines Telekommunikationsanbieters zur Installation eines Anschlusses zu einem Kunden vor Ort eingesetzt wird.

Kosten Ein weiterer nicht zu vernachlässigender Punkt sind die Kosten, z.B. für die Übertragung von Daten zu den am Prozess beteiligten Devices. Die Preise für mobile Datenanbindungen sind im Verhältnis zu kabelgebundenen Technologien wie Standleitungen, DSL oder im einfachsten Fall direkte LAN-Verbindungen immens teuer, weswegen bei der Verteilung von Prozessdaten, zum Beispiel für die Synchronisierung paralleler Aktivitäten bei der voneinander abhängiger Verwendung von Variablen in beiden Prozesszweigen darauf geachtet werden muss, welche Daten überhaupt übertragen werden sollen.

4.3 Herausforderungen mobiler Prozesse

Die oben aufgezählten Merkmale sowie eine Vielzahl weiterer Faktoren im Umfeld der mobilen Prozessausführung führen zu großen Herausforderungen, die beim Management und der Ausführung mobiler Prozesse auftreten. Neben den von außerhalb des Prozesses auftretenden Herausforderungen, wie physischer Verhinderung der Erreichung des Standorts ausführender Akteure, sind auch korrekte Kontroll- und Datenflüsse essentiell wichtige, relevante Herausforderungen, die beispielsweise bei der Ausführung paralleler Aktivitätspfade von Bedeutung sind.

Datenfluss In mobilen Prozessen, wie auch in herkömmlichen, nicht-mobilen Prozessen, ist es wichtig festzulegen, in welcher Art und Weise die im Prozess anfallenden Daten gespeichert und genutzt werden. So können Daten zum einen zentral auf einem Workflow-Management-Server gespeichert werden oder jeweils vom Vorgänger einer Aktivität an das zugeordnete Device der nachfolgenden Devices weitergereicht werden. Im Falle einer zentralen Datenhaltung treten Probleme im Zusammenhang mit der geringerer Zuverlässigkeit der Verbindungen der Devices auf, wenn diese einmal ausfallen sollte. Auch bei der

dezentralen Datenhaltung können bei Übermittlungsschwierigkeiten die gesamten oder nur Teile der zu übertragenden Daten verloren gehen. Eine weitere Herausforderung, die im Zusammenhang mit der dezentralen Datenhaltung beachtet werden muss ist, dass die meist geringe Kapazität der mobilen Devices und die hohen Kosten für die Übertragung der Daten eine wohlüberlegte Strategie erfordern, welche und wie viele Daten ausgetauscht werden sollen. Auch die Thematik der parallelen Ausführung von Aktivitäten bedarf einem besonderen Augenmerk, wobei hier eventuell auf geeignete Synchronisationsmechanismen Wert gelegt werden muss.

Kontrollfluss Ähnlich wie bei der Kontrolle des Datenflusses sind auch die Herausforderungen bei der Steuerung des Kontrollflusses eines mobilen Prozesses herausfordernd. So sind neben einer zentralen Steuerung, die häufig aufgrund der Konnektivität der Devices nicht geeignet ist, um die Bedürfnisse eines mobilen Prozesses zu handhaben, auch dezentrale P2P [24]-ähnliche „Steuerungen "oder Mischformen aus beiden Ansätzen denkbar. Unabhängig von der eingesetzten Technik, besteht die Herausforderung in jedem Fall im geeigneten Handling der unzuverlässigen Anbindung der eingesetzten Devices.

Robustheit Neben erforderlichen Konzepten bei der Handhabung des Daten- und Kontrollflusses ist die Robustheit der Ausführung ein sehr wichtiger Aspekt im Umfeld der mobilen Prozesse. Um eine robuste Ausführung der mobilen Prozesse zu gewährleisten, muss sichergestellt werden, dass der Prozess beim Auftreten unerwarteter Ereignisse nicht einfach abbricht oder sich verklemmt. So können durch mangelnde Konnektivität unter Umständen Daten nicht, oder nur stark verzögert, an das nachfolgende Device übertragen werden, was dazu führen kann, dass im Falle einer unzureichenden Fehlerbehandlung, die nachfolgenden Devices auf die vorhergehende Aktivität warten und sich der Prozess verklemmt. Auch die Übertragung beschädigter oder unvollständiger Daten kann zu großen Schwierigkeiten führen. Die defekten oder unvollständigen Daten müssen durch geeignete Ansätze, also beispielsweise einer wiederholten oder sogar erneuten Übertragung, durch die vollständige Ausführung der Vorgängeraktivität kompensiert werden können.

Allokation Eine weiterer Punkt, der eine Herausforderung im Umfeld mobiler Prozesse [16] darstellt, ist die Allokation, also die Zuweisung der Devices und/oder Benutzer zu einer Aktivität [33]. Je nach Ansatz wird hierbei zwischen der Zuordnung von Benutzern, die auf einem beliebigen Device verbunden sind, beziehungsweise der Annahme, dass Benutzer immer nur über ein zugeordnetes Device verbunden sind, was dann häufig vereinfachend als Host bezeichnet wird oder der „einfachen"Zuordnung von Devices zu Aktivitäten, unterschieden. Neben der Möglichkeit, die Allokation bereits zur Design-Zeit des Prozesses durch den Prozessdesigner modellieren zu lassen, kann die Zuordnung ebenfalls kurz nach der Instanziierung des Prozesses erfolgen. Auch eine flexible Zuordnung durch Analysieren der Umgebungsdaten und Errechnen der für eine Aktivität geeigneten Devices/Benutzer ist möglich, kann je-

doch auch durch interne Wissensdatenbanken, wie sie in [6] und [7] zum Einsatz kommen, erreicht werden. Hierbei kann die Bestimmung der geeigneten Nachfolger entweder durch ein zentrales System oder durch das mobile Device/den zugeordneten Benutzer selbst erledigt werden.

Herausforderung	**Hauptaspekte**
Datenfluss	zentrale Datenhaltung: geringe Zuverlässigkeit der Datenverbindungen dezentrale Datenhaltung: Beschädigungen / Verluste der Daten, geringe Kapazität der eingesetzten Devices, Synchronisierungen bei parallelen Ausführungen
Kontrollfluss	zentrale Steuerung, dezentrale Steuerung, P2P-ähnliche Mischformen: unzuverlässige Datenverbindungen kann Devices trennen
Robustheit	Prozess kann aufgrund unerwarteter Ereignisse (z.B. Deviceausfälle) verklemmen, beschädigte oder unvollständige Daten können zu falschen Ergebnissen führen und erfordern Behandlungsroutinenen
Allokation	Zuordnung von Benutzern, Rollen oder nur Devices, Zuweisung zur Designzeit, bei der Instanziierung oder dynamisch durch Berechnungen

Tabelle 2: Herausforderungen mobiler Prozesse

Nachdem in diesem Kapitel die Merkmale und Herausforderungen sowie die grundlegenden Eigenschaften mobiler Prozesse aufgezeigt worden sind, werden im folgenden Kapitel die Anforderungen, die aufgrund der Eigenschaften mobiler Prozesse an Workflow-/Prozess-Management-Systeme gestellt werden betrachtet und Ansätze zur Ausführung mobiler Prozesse dargestellt.

5 Anforderungen mobiler Prozesse an ein Workflow-Management-System

Mobile Prozesse stellen höhere Anforderungen an ein Workflow-Management-System als dies nicht-mobile (fixe) Prozesse tun. Aufgrund der eingesetzten Technologien, die, angefangen bei der Ausstattung der Geräte, über die immens höhere Gefährdung der Geräte im mobilen Umfeld, bis hin zu häufig unzuverlässigen Datenanbindungen, die allgemeine Zuverlässigkeit sehr verringern, bringt die Unsicherheit, an welchem Ort und mit welchen externen Ressourcen kommuniziert werden muss, große Anforderungen an die Flexibilität von Workflow-Management-Systemen im mobilen Umfeld mit sich.

Parallelität Bei mobilen Prozessen treten in der Regel vermehrt parallele Aktivitätsausführungen auf. Um mobilen Prozessen gerecht zu werden, muss ein

Workflow-Management-System ein hohes Maß an Parallelität bei der Aktivitätsausführung eines mobilen Prozesses unterstützen. Hierzu muss das System zum Beispiel Mechanismen zur Replikation von Prozessinstanzen oder deren Fragmenten sowie das Verteilen von Prozessdaten an mehrere Devices unterstützen.

Umgang mit Verbindungsproblemen Verbindungsabbrüche treten bei mobilen Prozessen nicht selten auf. So können Devices aufgrund der häufig recht unzuverlässigen Datenanbindungen nicht zu jeder Zeit eine Verbindung aufbauen oder verlieren bestehende Verbindungen, weil der ausführende Benutzer mit seinem Device in einen Bereich wechselt, wo er entweder keinen Empfang mehr oder eventuell keine Berechtigung für den Netzzugriff hat. Mit solchen Verbindungsproblemen müssen Workflow-Management-Systeme umgehen können, indem sie nachfolgende Aktivitäten warten lassen oder die durch einen längeren Verbindungsausfall blockierten Aktivitäten „umplanen", so dass diese auf anderen Devices erneut ausgeführt werden.

Konfigurationsunterstützung Das Workflow-Management-System sollte eine Untersützung der Konfigurationsmöglichkeiten und -aufgaben in hohen Maße zur Verfügung stellen. So sollte das System viele Möglichkeiten zur Zuweisung von Benutzern, Devices oder fest zusammen gehörigen Kombinationen von Benutzern und Devices, den sogenannten Hosts bieten. Auch zu den Benutzern beziehungsweise Devices selbst sollte das System etliche Daten verwalten können. Hier sind beispielsweise die Fähigkeiten der Benutzer interessant, um Ihnen Aktivitäten zuweisen zu können, die sie auch ausführen können. Im Falle der Devices kann es unter Umständen wichtig sein, deren Kapazität zur Speicherung von Daten zu kennen oder über verfügbare Sensoren wie Barcodescanner informiert zu sein.

Benutzer / Devices Prozessaktivitäten sollten in einem Workflow-Management-System nicht nur ausschließlich Devices zugeordnet werden können, sondern auch einzelnen Benutzern, die die Aktivität dann ausführen. Auch die Zuweisung von mehreren Benutzern zu einer Aktivität sollte möglich sein, damit beispielsweise im Krankheitsfall ein anderer Benutzer die Ausführung der Aktivität übernehmen kann und diese nicht den Prozessablauf blockiert (weil niemand sie ausführt). In bestimmten Fällen ist es auch durchaus denkbar, dass Benutzer- beziehungsweise Devicewechsel während der Aktivitätsausübung stattfinden müssen, was auch durch eingesetzte Workflow-Management-Systeme abgedeckt sein sollte.

Unterstützung sensorischer Daten Da die zur Aktivitätsausführung eingesetzten Geräte in den verschiedenen Einsatzfeldern mobiler Prozesse über Sensoren, wie beispielsweise Temperatursensoren, Drucksensoren, Sensoren zur Messung von Strömen oder auch Barcodescannern verfügen, müssen Workflow-Management-Systeme die Verarbeitung dieser sensorischen Daten unbedingt unterstützen.

Synchronisationsmechanismen Workflow-Management-Systeme sollen, wie bereits oben erwähnt, parallele Aktivitätsausführungen unterstützen. Durch diese parallelen Ausführungen ergeben sich bei beispielsweise in zwei parallelen Ausführungszweigen des Prozesses, bei Zugriff auf gleiche Variablen Abhängigkeitsprobleme, wenn eine der Aktivitäten die Variable lesen und abhängig vom Wert eine Entscheidung treffen muss, und die andere Aktivität diese schreiben muss. Auch eine beiden Teilzweigen nachfolgende Aktivität muss bei gleichen Variablen, die in mehreren parallelen Prozessteilen verändert wurden, durch ausgeklügelte Synchronisationsmechanismen den richtigen Wert für die weitere Ausführung des Prozesses festlegen können. Neben der Synchronisierung von Daten ist auch die Synchronisation des Prozesses nach parallel ausgeführten Aktivitäten essenziell.

Konfigurierbarkeit von Ablaufstrategien Im engem Zusammenhang mit der Fehlerbehandlung steht die Notwendigkeit, Ablaufstrategien konfigurieren können zu müssen. Bei wie auch immer gearteten Schwierigkeiten sollte es im Workflow-Management-System möglich sein, Strategien zu konfigurieren, die in diesem Falle die Situation entschärfen und den aufgetretenen Fehler kompensieren, um ein Verklemmen des Prozesses zu verhindern.

Monitoring Die Überwachung von Prozessen ist immens wichtig, um einerseits bei Fehlern passende Ablaufstrategien zu deren Kompensation zu fahren. Andererseits ist das Monitoring auch im normalen Betrieb des Workflow-Management-Systems eine wichtige Funktion, um den Prozessfortschritt zu überwachen oder nachträglich Schwachstellen der Prozessausführung zu erkennen, die dann durch eine Überarbeitung des Prozesses beseitigt werden können.

Die wichtigste Forderung an ein Workflow-Management-System (WfMS) im mobilen Umfeld ist die verteilte Ausführung des Prozesses. Ein WfMS muss, um die einleitend geforderte Flexibilität der Prozesse zu gewährleisten, eine verteilte Ausführung einzelner Prozessaktivitäten bzw. ganzer Prozessfragmente auf den Devices ermöglichen. Dabei ist es unerheblich, ob ein Device eine feste kabelgebundene Arbeitstation oder ein mobiles Gerät ist. Die Übergabe der Aktivitäten oder Fragmente muss jedem Fall durch das WfMS möglich sein. Aufgrund der Wichtigkeit dieser Anforderung an ein WfMS werden im folgenden Ansätze, der für uns relevanten Möglichkeiten der physischen Fragmentierung und der logischen Fragmentierung von verteilen Prozessen, näher betrachtet.

5.1 Physische Fragmentierung

Unter der physischen Fragmentierung (Abbildung 6) von Prozessen versteht man im Allgemeinen das Aufteilen eines großen Prozesses in mehrere kleinere Prozessfragmente. Diese werden nach der Allokation der Prozessfragmente

an die entsprechenden Prozess-Engines zur Ausführung der Fragmente verteilt. Hierbei erfolgt die Fragmentierung immer statisch. Diese kann entweder während der Designzeit durch den Designer des Prozesses, direkt nach der Instanziierung des Prozesses oder spätestens nach der Ausführung der Startaktivität erfolgen. Bei langlaufenden Prozessen kann dies gegebenenfalls gegen den Einsatz der physischen Fragmentierung sprechen, da aufgrund der statischen Zuweisung, Änderungen im Prozess nur schwer möglich sind.

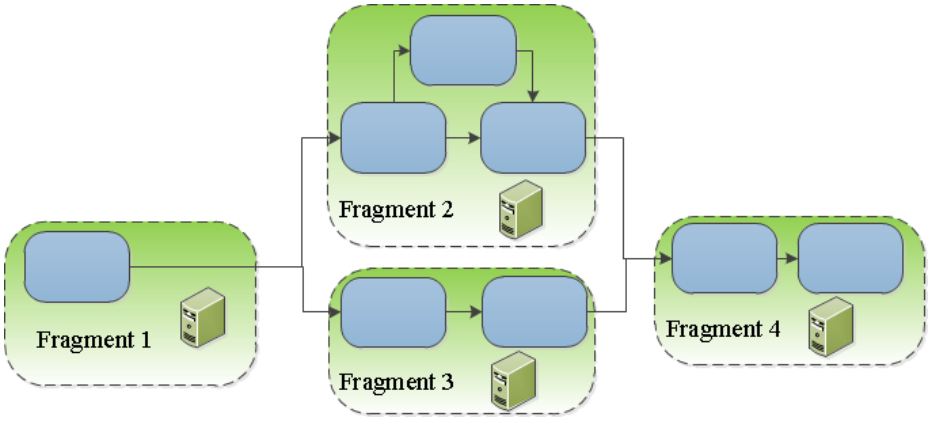

Abbildung 6: Physische Fragmentierung im Überblick

Der Einsatz eines zentralen WfMS ist bei der physischen Fragmentierung zumeist nicht vorgesehen, da nach der Verteilung der Prozessfragmente auf die ausführenden Prozess Engines keine Kommunikation mit der verteilenden Instanz mehr vorgesehen ist. Somit ist eine Steuerung oder Überwachung durch eine zentrale Instanz häufig ungünstig oder schlicht nicht möglich. Durch den Wegfall eines zentralen WfMS ist auch die zentrale Speicherung von Prozessdaten nicht möglich. Daher werden Prozessdaten nur zwischen den beteiligten Fragmenten ausgetauscht, was bei Ausfällen sehr problematisch sein kann.

Ein wichtiger Aspekt der physischen Fragmentierung ist, dass der ursprüngliche Prozess immer verändert wird, um den Gegebenheiten der Ausführungsumgebung angepasst werden zu können. Zudem haben die beteiligten Devices in der Regel keine Kenntnis vom ursprünglichen Prozess und kennen nur die Schnittstellen zu anderen Fragmenten, deren Vorgänger bzw. Nachfolger sie sind und mit denen sie kommunizieren.

Der nächste Abschnitt zeigt wichtige Ansätze zur physischen Fragmentierung auf. Abschließend werden diese auf ihre Eignung für den Einsatz zu mobilen Prozessausführung im Verständnis dieser Arbeit bewertet.

Ansätze Die folgenden vier Ansätze stellen jeweils eine Möglichkeit dar, wie die physische Fragmentierung von (mobilen) Prozessen umgesetzt werden kann.

1: CiAN: A Workflow Engine for MANETs (Collaboration in Ad hoc Networks) [6] ist eine Workflow-Engine für *MANET*s. Unter einem *MANET* versteht man ein mobiles ad-hoc Netzwerk, das mehrere Devices zu einem Netz verbindet und die sich selbst konfigurieren können, so dass für den Betrieb des *MANET*s keine zentrale Serverinstanz notwendig ist (Abbildung 7). In CiAN wird die vereinfachende Annahme getroffen, dass jeder Benutzer, der an der Ausführung einer Aktivität beteiligt ist, ein ihm zugewiesenes mobiles Device besitzt und dies stets mit sich führt. Daher unterscheidet *CiAN* im Wesentlichen nicht zwischen den Benutzern und den Devices, sondern spricht in diesem Zusammenhang zusammenfassend von den Hosts. Ein Host stellt anderen Hosts Informationen über sich selbst zur Verfügung. Auf die Zuordnung der Hosts zu Aktivitäten wird später eingegangen. Neben einem eindeutigen Namen, sind dies ein Plan, zu welchen Zeiten des Host bereits mit anderen Aktivitäten belegt ist und eine Liste der angebotenen Dienste des Hosts. Benötigt ein Host einen externen Dienst, so kann er diesen über eine integrierte SOAP [36]-Schnittstelle abrufen.

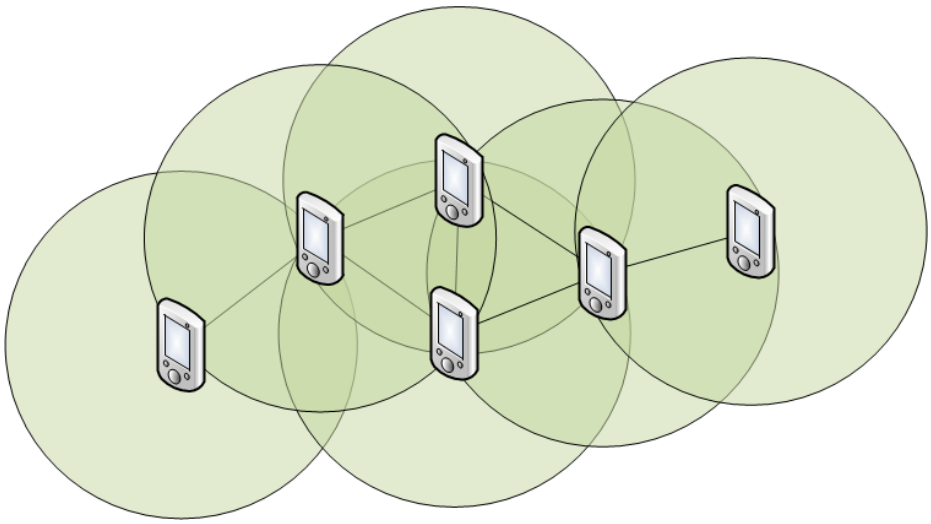

Abbildung 7: Mobiles ad-hoc Netzwerk

In *CiAN* versteht man unter eine Choreographie die Allokation der Aktivitäten zu den Hosts, die aber wie bereits oben erwähnt wurde, nicht näher betrachtet wird, wohingehen die *CiAN*-Engine für die Umsetzung der für den Nachrichtenaustausch benötigten Protokolle zuständig ist. Um dieser Teilung Rechnung zu tragen, arbeitet *CiAN* in zwei unterschiedlichen Modi, dem Planungsmodus (Abbildung 8) und dem Arbeitsmodus (Abbildung 9), wobei der Planungsmodus nur auf einem sogenannten Planungs-Host ausgeführt wird und alle anderen Host sich im Arbeitsmodus befinden.

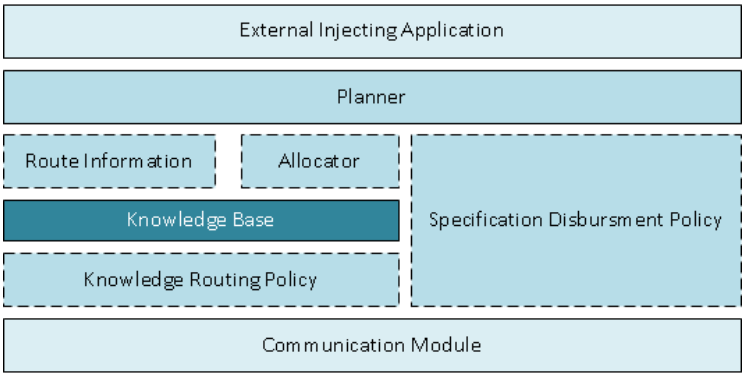

Abbildung 8: Komponenten des Planungsmodus [6]

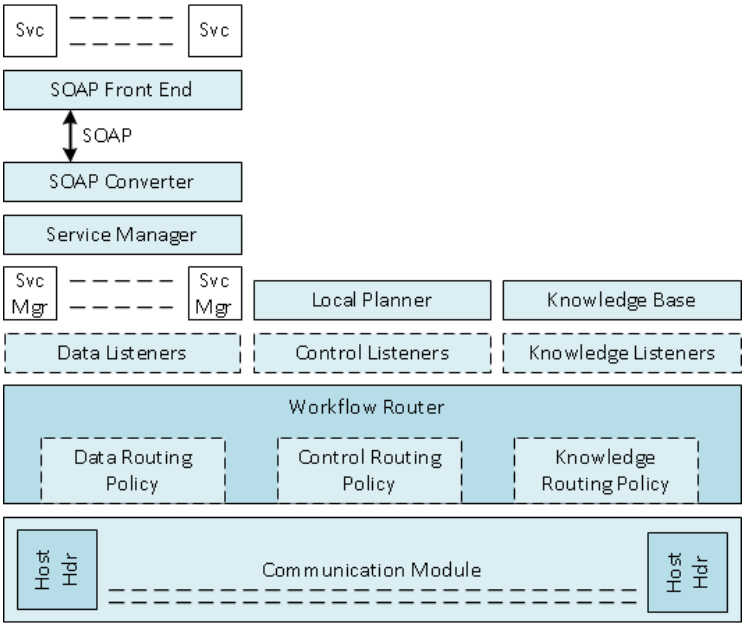

Abbildung 9: Komponenten des Arbeitsmodus [6]

Im Folgenden wird der grundlegende Ablauf der Ausführung eines Workflows / Prozesses in *CiAN* erläutert. Ein bereits bestehender Workflow muss für die Ausführung über *CiAN* als Graph vorliegen oder in einen Graphen konvertiert werden, wobei die verwendete Standard-Notation von den Autoren der Arbeit nicht erwähnt wird. Die Erstellung des Workflows sowie die Allokation der Hosts werden durch *CiAN* nicht näher spezifiziert. Im Fall der Allokation der Hosts verweisen die Autoren auf einen Beispielalgorithmus aus einer anderen Arbeit. Zu Beginn sind alle Hosts benachbart. Die Kommunikation der Hosts erfolgt hierbei dezentral über WLAN 802.11 b/g/n [25], sobald sich zwei Hosts in Reichweite befinden. In dieser initialen Phase werden Informationen

über alle beteiligten Hosts untereinander ausgetauscht, wodurch jeder Host zu Beginn der Prozessausführung Informationen über jeden Host besitzt. Der Planungshost, der nach dem Austausch der Host-Informationen über die angebotenen Dienste und die Zeiten in denen ein Hosts nicht verfügbar ist Bescheid weiß, teilt den restlichen Hosts die Aktivitäten zu, woraufhin der Prozess gestartet werden kann. Diese Zuteilung erfolgt automatisch unter Anwendung eines Allokations-Algorithmus durch den Planungshost, kann aber auch verteilt durch mehrere Planungshosts erfolgen.

Nach dem Aktivieren der Start-Aktivität werden die Folgeaktivitäten, deren Vorbedingungen erfüllt sind, ausgeführt. Sind nicht alle Vorbedingungen für die Aktivitätsausführung auf einem Host erfüllt, so wartet dieser auf eingehende Daten des Vorgängers. Um Daten im Ablauf des Prozesses austauschen zu können, werden diese durch ein sogenanntes *„gossiping "* - Protokoll ausgetauscht, d.h. die Informationen werden von Host zu Host weitergereicht und verbreiten sich so über alle beteiligten Hosts. Für weiterführende Informationen zu *„gossiping "*verweise ich an [9]. Die Ausführung endet mit dem Abschluss der Ende-Aktivität. Eine Fehlerbehandlung wird von den Autoren nicht aufgegriffen, soll aber in folgenden Arbeiten behandelt werden.

2: MobiWork: Mobile Workflow for MANETs [7] ist eine Workflow-Engine die auf *MANET*s arbeitet. Aufgrund der Verwendung von *MANET*, die bereits im vorhergehenden Kapitel erläutert wurden, entfällt die Notwendigkeit einer zentralen Serverinstanz zur Verwaltung des Workflows. Ähnlich wie beim *CiAN*-Ansatz, wird auch bei *MobiWork* die vereinfachende Annahme getroffen, dass ein Device fest einem Benutzer zugewiesen ist. Zusätzlich ist jeder Host jedoch in der Lage, mehrere Softwareinstanzen auszuführen, die sogenannten Agents, was jedoch in der vorliegenden Arbeit auf einen einzelnen Agent pro Host eingeschränkt wird. Agents haben in diesem Ansatz nichts mit den bereits eingangs agentenbasierten Ansätzen verteilter Prozesse gemein, sondern bezeichen im Kontext dieses Ansatzes lediglich Softwareinstanzen. Hosts besitzen in *MobiWork* eine eindeutige *ID*, eine Liste seiner Qualifikationen und die maximale Geschwindigkeit mit der sich ein Host bewegen kann. Die maximale Bewegungsgeschwindigkeit ist bei der Allokation von Aktivitäten von Bedeutung, da durch sie festgestellt werden kann, ob ein Host dazu in der Lage ist, sich zwischen zwei Aktivitätsausführungen an den Ort der folgenden Aktivitätsausführung zu bewegen, um dort die Folgeaktivität zu starten.

In *MobiWork* wird eine Choreographie nicht explizit erwähnt, jedoch nimmt ein Planungsmodul (Abbildung 10) diese Aufgabe war, indem es unter Zuhilfenahme eines Allokationsalgorithmus Aktivitäten zu den Hosts zuordnet und diese an die entsprechenden Hosts verteilt. In der Arbeit werden zwei Allokationsalgorithmen beschrieben. Zum einen ist dies ein naiver Algorithmus, der pro Aktivität eine Tabelle von zuordbaren Hosts erzeugt, die auch durch Zuweisungen entstehende Konflikte enthält. Diese Tabellen werden nach ihrer Berechnung durchiteriert, um eine gültige Allokation für den gesamten Plan

zu bekommen. Der zweite Algorithmus ist eine erweiterte Variante des naiven Algorithmus, der durch Rekursion und die Anwendung eines Stacks die Performanz des ersten Algorithmus steigert.

Die Planungskomponente ist in *MobiWork* in jedem Host vorhanden. Das Vorhandensein der Komponente liegt darin begründet, dass bei Auftreten von Fehlern bei der Ausführung von Aktivitäten auf dem betroffenen Host ein sogenanntes Re-Planning stattfindet, um eine erneute Ausführung oder Re-Allokation der Aktivität, abhängig vom Grad der Schwere eines Fehlers, durchführen zu können.

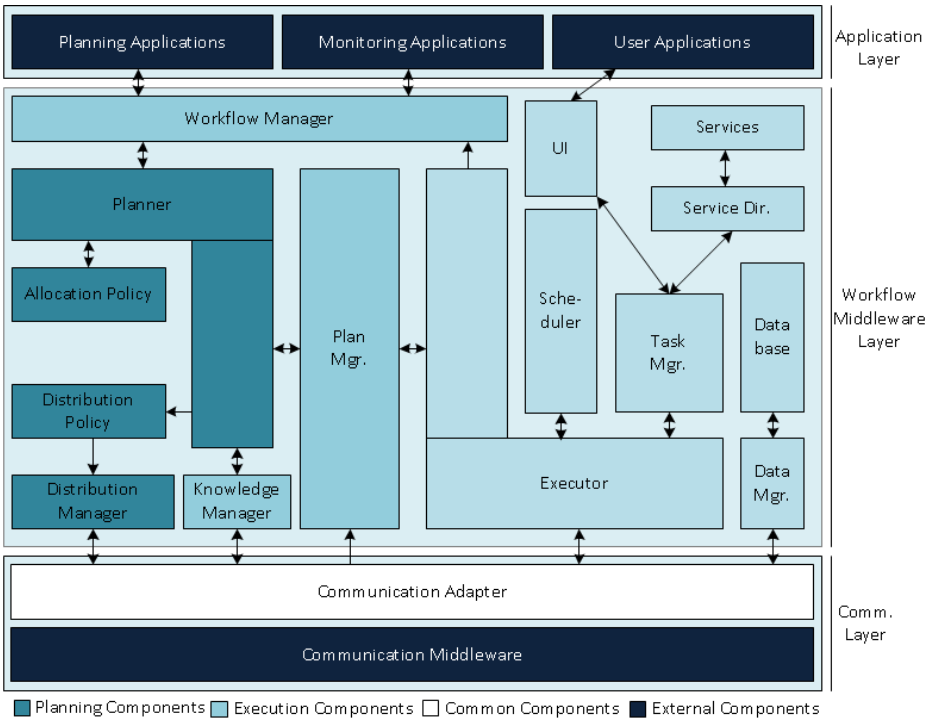

Abbildung 10: Architektur der MobiWork-Engine mit Planungskomponenten [7]

Der grundlegende Ablauf eines Workflows in *MobiWork* ähnelt dem in *CiAN*. Nach einer initialen Phase in der alle Hosts benachbart sind, wird der Workflow auf den Host des *Group Leaders* geladen. Die Planungskomponente des Hosts übernimmt nun die automatische Allokation der Aktivitäten und verteilt diese unter Zuhilfenahme einer Fragmentierungs-Komponente an die anderen Hosts, die dann nur ihre Aktivitäten sowie deren Vorgänger und Nachfolger mit den zu empfangenden/ zu sendenden Daten kennen. Der *Group Leaders* kann bei *MobiWork* den Workflow monitoren, muss jedoch unter Umständen aufgrund des für *MANET*s typischen „gosipping“-Protokolls zur Verteilung aktualisierter Informationen von Hosts oder Aktivitäten mit veralteten

Informationen rechnen.

Der Workflow, den der *Group Leaders* zu Allokation und Verteilung der allokierten Aktivitäten bekommt, muss in *MobiWork* als gerichteter Graph vorliegen, der sogenannte Plan. Hierbei stellen die Kanten die Ordnung des Workflows dar und die Knoten die Aktivitäten. Jeder Knoten verfügt über eine *ID*, die für die Aktivität benötigte Fähigkeiten eines ausführenden Hosts, Ort, Start- und Endzeit der Ausführung sowie die Vorgänger und Nachfolger mit den zu empfangenden bzw. weiterzureichenden Daten. Zusätzlich hat jeder Konten einen zugeordneten Host und einen Zustand, der den Grad der Fertigstellung der Aktivität abbildet.

3: From Centralized Workflow Specification to Distributed Workflow Engine beschreibt ein Modell zur verteilten Ausführung von Workflows, das in *Mentor* (Middleware for Enterprise-wide Workflow Management) [8] eingesetzt wird. *Mentor* ist ein auf dem Client-Server-Prinzip beruhendes Workflow-/Prozess-Management-System zur Ausführung von umfangreichen Workflows/Prozessen, die über ganze Unternehmen verteilt sein können.

Für die Darstellung der Workflows werden in *Mentor* sogenannte *State-* (Abbildung 5) und *Activity-Charts* (Abbildung 6) eingesetzt. Diese basieren auf einem formalen Modell mit strikter Semantik, das eine automatische Fragmentierung erlaubt.

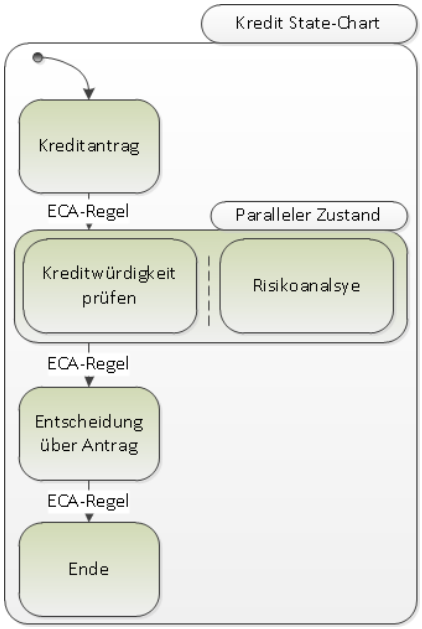

Abbildung 11: State Chart Kredit-Antrag-Prozess

State-Charts bilden dabei das Verhalten des Systems ab und sind eine erweiterte Form von Zustandsdiagrammen, mit einem festgelegtem Startzustand und sogenannter ECA - Regeln (Event-Condition-Action) an den Kanten, die Bedingungen für den Übergang zwischen den Zuständen darstellen. *Activity-Charts* bilden die Funktionalität eines Workflows ab, wobei die aktiven Komponenten des Diagramms direkt den Aktivitäten des Workflows und den Zuständen des *State-Charts* entsprechen.

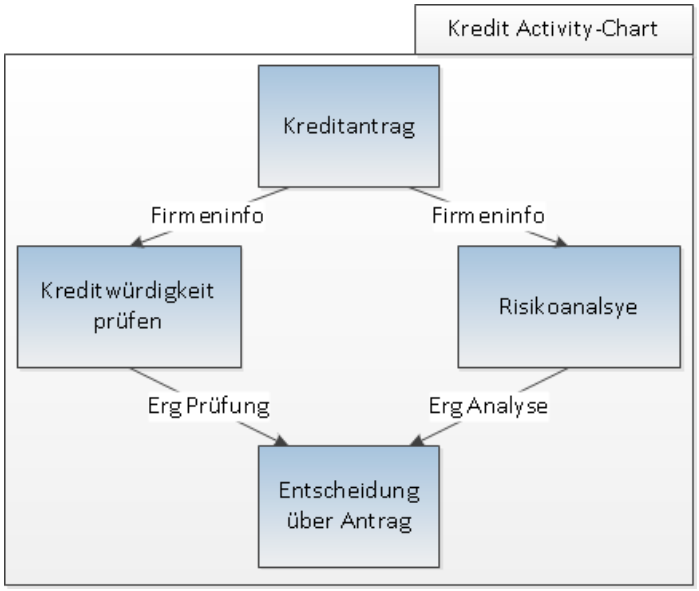

Abbildung 12: Activity Chart Kredit-Antrag-Prozess

Bei der Fragmentierung des Prozesses müssen sowohl die *Activity-* als auch die *State-Charts* fragmentiert werden. *Mentor* geht davon aus, dass jeder Aktivität bereits eine zugehörige Ausführungs-Rolle zugewiesen wurde, weswegen die *Activity-Charts* recht einfach anhand identischer Rollen fragmentiert werden können. Die Fragmentierung der *State-Charts* ist hingegen aufwändiger, da diese nicht durch eine einfach Zuweisung von Zuständen zu den Fragmenten funktioniert. Nach der Zuweisung der Zustände zu den Aktivitäten, müssen die *State-Charts* orthogonalisiert werden, und können dann den Workflow-Servern zugewiesen werden. Für weitergehende Informationen über die Orthogonalisierung der *State-Charts* wird an dieser Stelle auf [8], Kapitel 5 verwiesen.

Da für die verteilte Ausführung eines *State-Charts* nach jeder ausgeführten Aktivität, das Aktualisieren der Zustände in allen beteiligten Fragmenten notwendig ist, wird dies über den Austausch von sogenannten Synchronization Messages gelöst. Diese Vorgehensweise wird als *Strict Synchronization* bezeichnet, da sie verlangt, dass ein instanziiertes *State-Chart* nur dann synchronisiert ist, wenn alle möglichen Paare untereinander den jeweils vollständigen

Systemzustand ausgetauscht haben. Dies kann zu einer großen Anzahl an auszutauschenden Nachrichten führen, weswegen die Autoren zusätzlich mehrere aufgeweichte Synchronisationsdefinitionen angeben, die die Menge der übertragenen Daten reduzieren. Um die Anzahl den Nachrichten zu verringern, führen die Autoren ein weiteres Synchronisierungsschema ein, die *Weak Synchronisation*, bei der Fragmente mehrere Aktivitäten nacheinander ausführen dürfen, ohne nach jedem Schritt eine Synchronisierung vornehmen zu müssen. Somit ist die Synchronisation zweier Fragmenten von den anderen Paaren unabhängig.

Die eigentliche Ausführung der Workflows geschieht in *Mentor* durch Verteilung des Workflows auf mehrere modular aufgebaute Workflow-Server (Abbildung 13), die untereinander über Transitionsmonitore Nachrichten über den aktuellen Zustand austauschen. Die Ausführung der Aktivitäten erfolgt dann auf den beteiligten Clients.

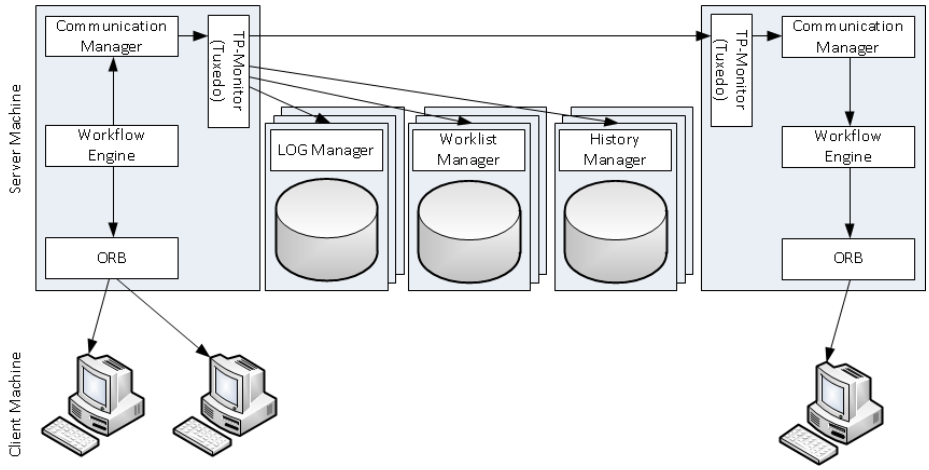

Abbildung 13: Architektur des Mentor-Ansatzes [8]

4: „A Novel Approach to decentralized Workflow Enactment“ [10] beschreibt einen Ansatz zur Erweiterung der bisherigen, zentralisierten Ausführungsansätze in herkömmlichen WfMS um die Möglichkeit die Prozesse dezentralisiert ausführen zu können. Der Ansatz verzichtet auf den Einsatz einer zentralen, steuernden Serverinstanz, verlangt keine Choreographie und benötigt keine Re-Design des originalen Prozesses. In der Arbeit wird sogar ausdrücklich hingewiesen, dass Änderungen am modellierten Prozess ausschließlich aufgrund von notwendigen Änderungen am zugrundeliegenden Geschäftsprozess erfolgen sollten. Die Merkmale von Benutzern beziehungsweise Hosts werden im Ansatz nicht diskutiert, weswegen bei der Allokation von Aktivitäten zu Devices nur allgemeine Vorgehensweisen knapp angedeutet werden und von Prozessteilnehmern gesprochen wird. Der nächste Absatz beschreibt die von den Autoren angegebene Vorgehensweise von einem kompletten Prozess bis zur

Ausführung der verteilten Fragmente.

Der ersten Schritt ist das Design des kompletten Prozesses. Die Autoren betonen hier, dass der Vorgang des Designs nicht durch ihren Ansatz verändert oder beeinträchtigt wird, sondern wie gewohnt mit herkömmlichen Werkzeugen geschieht. Die einzige Forderung an den Designer ist, dass nach diesem Schritt ein gültiger Prozess in BPEL [26] vorliegt.

Der nächste Schritt ist das Fragmentieren des Prozesses, worunter ein Aufsplitten des Prozesses auf die einzelnen Prozessteilnehmer verstanden wird, was den Autoren nach zum Beispiel anhand der physischen Infrastruktur geschehen kann. Hier werden die Prozessmodelle zusätzlich über Informationen angereichert, dem sogenannten *Deployment Descriptor*, der eine Zuweisung zu den Prozessteilnehmern ermöglicht. Dabei werden als Alternativen für die Fragmentierung des Prozesses zwei Alternativen, die manuelle und die automatische Fragmentierung des Prozesses angegeben. Für die automatische Fragmentierung des Prozesses nennen die Autoren Möglichkeiten, beispielsweise anhand von Daten der Prozessteilnehmer, wie etwa deren verfügbare Ressourcen, oder durch das Lösen von Optimierungsproblemen (Lokale Suche, Hill-Climbing oder Simuliertes Abkühlen) über die Kostenfunktion (Ausführungskosten, Netzdistanz, Anzahl benötigter Ausrufe, Ausführungswahrscheinlichkeiten, Variablengröße, etc.) die Aufteilung des Prozesses vorzunehmen.

Nach der Fragmentierung des Prozesses erfolgt eine Transformation des BPEL-Prozesses in einen von den Autoren selbst entwickelten Dialekt der Petri-Netze, den Executable Workflow Networks (EWFN), der speziell dafür entwickelt wurde, um auf *Tuplespaces*, die in [11] detailliert beschrieben werden, nativ ausgeführt werden zu können und der den in den *Tuplespaces* eingesetzen Mechanismus zur Weitergabe von *Tokens* formalisiert. Die Autoren geben hierzu in Kapitel 3 eine formale Definition an und beschreiben den Vorgang der Transformation detailliert in Kapitel 4.

Als letzten Schritt werden die Fragmente des Prozesses auf die Prozessteilnehmer deployed und dann ausgeführt, indem das Prozessmodell durch Rufen einer *Pick-* oder *Receive-Aktivität* instanziiert wird. Die einzelnen Aktivitäten kommunizieren durch die Weitergabe von Tokens untereinander und teilen Prozessdaten/-variablen durch das Speichern und Abrufen von *Tuples* auf dem gemeinsamen *Tuplespace*. Hierbei können Aktivitäten Variablen speichern (*write*), destruktiv holen (*take*) oder lesen (*read*). Eine direkte Kommunikation der Prozessteilnehmer entfällt somit, weswegen der *Tuplespace* auch als möglicher Flaschenhals Probleme bereiten kann.

Zwischenfazit Bei näherer Betrachtung der physischen Fragmentierung fällt auf, dass diese in Verbindung mit mobilen Prozessen im Allgemeinen nicht die nötige Flexibilität bietet, die für die mobile Unterstützung von Prozessen von

WfMS verlangt wird. In allen vier oben gezeigten Ansätzen erfolgt die Festlegung, welcher Prozessteilnehmer die Aktivität ausführen soll statisch, entweder zur Designzeit oder während der Instanziierung des Prozesses. Später notwendige Änderungen können von dieses Modellen nur unzureichend berücksichtigt werden.

Benutzerzuweisungen oder sogar Wechsel werden in den obigen Ansätzen nicht betrachtet oder zur Vereinfachung der Einsatzszenarien einfach weggelassen. Da gerade im mobilen Prozessumfeld Benutzerzuweisungen und -wechsel von großer Bedeutung sind, ist es erforderlich, dass die Autoren der Ansätze in zukünftigen Weiterentwicklungen mehr Aufmerksamkeit auf diese Thematik lenken. Viele Ansätze vernachlässigen auch die Behandlung von Fehlern, die während der Prozessabläufe auftreten können, weswegen die Robustheit der Ansätze nicht beurteilt werden kann.

Der Hauptpunkt, der die physische Fragmentierung von verteilten Workflows nach Betrachtung der vier Ansätze als unzureichend geeignet für die mobile Prozessausführung kennzeichnet, ist die Unkenntnis der Prozessteilnehmer über den vollständigen Prozess. Durch diese Unkenntnis ist es im Fehlerfall nur sehr schwer möglich, durch die dezentralisierte Ausführung der Fragmente, passende Deeskalationsstrategien zu finden. Im folgenden Abschnitt wird eine andere Möglichkeit der verteilten Prozessausführung diskutiert, die den Punkt der Unkenntnis über den vollständigen Prozess, berücksichtigt.

5.2 Logische Fragmentierung

Ein weitere Möglichkeit Prozesse verteilt auszuführen, ist die logische Fragmentierung [1] von Prozessinstanzen. Der Unterschied zu der physischen Fragmentierung ist, dass bei der logischen Fragmentierung von Prozessen der gesamte Prozess mit seinem Daten- und Kontrollfluss vollständig von einem Device, das eine Aktivität ausgeführt hat, an den nachfolgenden Akteur übergeben wird und dieser somit über den gesamten Prozesszustand Kenntnis hat. Dies wird erreicht durch die Migration der Prozessinstanz, und es ist somit auch nicht nötig, den originalen Prozess physisch in mehrere Fragmente aufzuteilen. Daher bleibt bei dieser Art der verteilten Prozessausführung der originale Prozess, wie er von einem Workflow-Designer entworfen wurde, in den meisten Fällen strukturell unangetastet und wird nicht verändert. Aus diesem Grund kann man den Fragmentierungsvorgang bei der Prozessmigration auch als logische Fragmentierung bezeichnen [1], da nur die Verantwortlichkeiten für die Ausführung der Aktivitäten in Fragmente aufgeteilt werden.

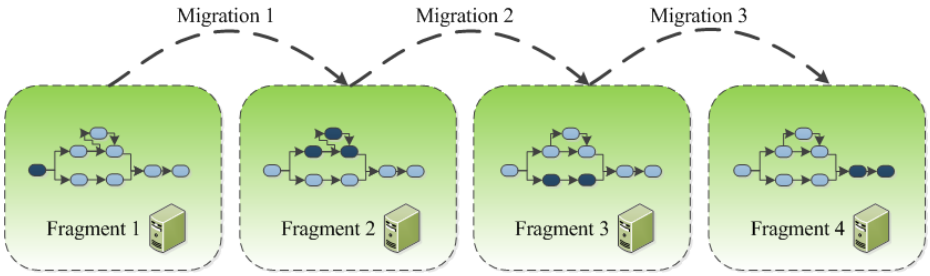

Abbildung 14: Logische Fragmentierung im Überblick

Durch die vollständige Migration der Prozessinstanz ergeben sich jedoch auch Probleme. Da jeder Teilnehmer am Prozess über den vollständigen Zustand und den gesamten Ablauf des Prozesses durch die Migration Kenntnis erlangt, ist es schwierig bei sicherheitsrelevanten Teilen des Prozesses durch geeignete Sicherheitsmechanismen dafür zu sorgen, dass nur Befugte Wissen darüber erlangen oder er nicht von Unbefugten korrumpiert wird.

Zusätzlicher Aufwand fällt bei der Prozess Migration für die Abbildung der Fragmente an. Aufgrund der Forderung den originalen Prozess unangetastet zu lassen, ist es nötig, die Informationen zusätzlich zum Prozessmodell, durch sogenannte Migrationsdatenmodelle zur Verfügung zu stellen. So ist es möglich, diese als Bemerkungen im originalen Prozessmodell anzuheften, sollten jedoch wenn möglich, außerhalb des Prozessmodells als beschreibendes Zusatzdokument mit dem Prozess weitergereicht werden.

Ein weiteres Merkmal in diesem Zusammenhang hat positive und negative Eigenschaften. Da die Prozessinstanz im Ganzen migriert, ohne physisch fragmentiert zu werden, entfällt hierbei die Notwendigkeit der Koordination und Zusammenführung von Prozessfragmenten, wenn diese sequentiell ausgeführt werden [1]. Im Gegenzug ist bei paralleler Ausführung von Aktivitäten mehr Aufwand nötig, da die Prozessinstanz ja nicht physisch fragmentiert wurde. Um nun parallele Ausführung zu ermöglichen, ist es nötig, mit Replikaten der Prozessinstanz zu arbeiten, die wiederum, nach Zusammenführung des Kontrollflusses auf eine einzelne Nachfolgeaktivität, synchronisiert werden müssen.

Positiv ist im Zusammenhang mit der Migration zu werten, dass eine logische Fragmentierung, also die Allokation der Aktivitäten zu den am Prozess teilnehmenden Devices im Gegensatz zur physischen Fragmentierung, die Devices immer statisch zur Designzeit oder während der initialen Phase des Prozesses zuweist, auch zur Laufzeit des Prozesses zugewiesen werden können, was die große Flexibilität des Migrationsansatzes aufzeigt.

Durch die Migration der gesamten Prozessinstanz sind für die Ausführung eines verteilten Prozesses keine zentralen Server nötig. Daraus ergibt sich aus

Sicht dieser Arbeit ein weitere großer Pluspunkt für den Einsatz von Migrationsansätzen im Umfeld der mobilen Prozesse. Da die mobilen Devices, wie bereits oben erwähnt, häufig nur über wenig performante und nur unzureichend stabile Datenverbindungen verfügen, wodurch der Kontakt zu zentralen Servern erschwert wird, ist der Einsatz zentraler Server hier ungeschickt oder technisch aufwendig, um dieses Defizit zu kompensieren. Durch die Migration entfällt aber diese Abhängigkeit von einem zentralen Server, weswegen sich dieser Ansatz ausgezeichnet für mobile Prozesse eignet.

Im folgenden Abschnitt werden für uns relevante Aspekte der Klassifikation von Ansätzen zur logischen Fragmentierung detailliert betrachtet.

Klassifikationsmerkmale Neben den der logischen Fragmentierung zugrunde liegenden Merkmalen, die im vorherigen Abschnitt besprochen wurden, ist es aus Gründen der Vergleichbarkeit der unterschiedlichen Ansätze notwendig, diese anhand der Ausprägungen der Merkmale einordnen zu können. Die nächsten Punkte bieten eine detaillierte Beschreibung, der für diese Arbeit relevanten Klassifikationsmerkmale.

Architektur Im vorherigen Abschnitt wurde die Behauptung aufgestellt, dass bei Ansätzen zur logischen Fragmentierung, aufgrund der verteilten Ausführung der Prozesse, keine zentrale Serverinstanz benötigt wird. Obwohl dies auf die meisten Ansätze, die den Ansatz zur logischen Fragmentierung verfolgen, zutrifft und diese vollständig ohne einen zentralen Server zur Verwaltung der Prozesslogik auskommen, existieren auch Ansätze, die einen solchen Server vorsehen.

Daten Auch anhand der Deklaration der Daten lassen sich unterschiedliche Ansätze klassifizieren. So kann es von Interesse sein, ob Ansätze zur logischen Fragmentierung neben globalen Variablen auch lokale Variablen vorsehen, die dann nur innerhalb einer Aktivität beziehungsweise einem Fragment Gültigkeit besitzen und nach der Beendigung der Aktivität oder der letzten Aktivität eines Fragments verworfen werden. Ein weiterer Aspekt im Bezug auf die Daten bei Ansätzen zur logischen Fragmentierung ist die Art der Datenhaltung. So können Daten zum einen auf einer zentralen Datenbank gespeichert werden oder aber auch direkt mit der Prozessinstanz von Prozessteilnehmer zu Prozessteilnehmer weitergegeben werden.

Migrationsentscheidung Bei der logischen Fragmentierung können verschiedene Ansätze der Entscheidung, durch was eine Migration veranlasst wird, klassifiziert werden. Wie bei der physischen Fragmentierung ist es auch bei der logischen Fragmentierung möglich, Aktivitäten statisch während der Designzeit festzulegen und dadurch im Falle z.B. eines Device-Wechsels eine Migration zu veranlassen. Jedoch existieren bei der logischen Fragmentierung noch weitere Möglichkeiten, Migrationen auszulösen. Eine Möglichkeit besteht darin,

der Prozess-Engine, die die aktuelle Aktivität ausgeführt hat, die Entscheidung darüber zu überlassen , ob und wohin eine Migration stattfinden soll. Auch algorithmische Bestimmungen, zum Beispiel anhand der Prozessparameter oder aus Gründen der Servicequalität sind hier möglich [1].

Prozessstruktur Wie schon bei der Architektur, so muss nun auch bei der Prozessstruktur von der Behauptung abgerückt werden, dass der originale Prozess immer unangetastet bleiben sollte und somit sein Zustand unverändert bleibt. Im Falle, dass die Migrationsdaten, die die Fragmentierung des Prozess abbilden, als weiteres Dokument mit der Prozessinstanz migriert werden, bleibt der ursprüngliche Prozess auch unverändert erhalten. Ist ein solches Zusatzdokument allerdings nicht vorgesehen, muss ein anderer Mechanismus gefunden werden, um die logische Fragmentierung abzubilden. Hier werden die Migrationsdaten in den ursprünglichen Prozess eingebunden, indem zum Beispiel Migrationsaktivitäten eingefügt werden [1].

Mächtigkeit der mobilen Engine Neben den bisher erwähnten Klassifikationsmerkmalen ist auch die Mächtigkeit der mobilen Engine ein betrachtenswertes Kriterium. Um auf einem (mobilen) Device eine Aktivität oder ein Fragment, das heißt größere Prozessfragmente, auszuführen, muss auf dem Device eine Engine verfügbar sein. Hier kann unterschieden werden, ob die eingesetzte Engine nur recht simpel ist und nur einzelne Aktivitäten ausführen kann oder ob eine mächtigere Engine eingesetzt wird. Umfangreichere mobile Workflow-Engines sind in der Lage größere Teilprozesse auszuführen, wodurch zum Beispiel unnötige Migrationsvorgänge vermieden werden können, da nicht nach jeder Aktivität migriert werden muss.

Ansätze Die folgenden fünf Ansätze stellen jeweils eine Möglichkeit dar, wie die logische Fragmentierung von (mobilen) Prozessen umgesetzt werden kann.

1: Flexible Execution of Distributed Business Processes based on Process Instance Migration [1] Die Autoren der stellen einen Ansatz zur verteilten Ausführung von Prozessen durch logische Fragmentierung des Prozesses vor. Um das originale Prozessmodell nicht zu verändern, wird in diesem Ansatz der Einsatz von Migrationsdatenmodellen als eigenständiges Zusatzdokument vorgesehen, das den Ort und Zeitpunkt der Migration des Prozesses durch die Erweiterung der Prozessbeschreibung um Prozess- und Aktivitätszustände vorgeben kann. Auch Zuweisungen beziehungsweise Verfahren zur Festlegung des Prozessteilnehmers, der den auf den aktuellen folgenden Prozessschritt ausführen soll, können mit diesen Modellen festgelegt werden. Um ihr Migrationsdatenmodell einsetzen zu können, setzen die Autoren voraus, dass die Prozessbeschreibung atomare Aktivitäten und zusammengesetzte Aktivitäten, also ganze Prozessfragmente, sowie die Deklaration von lokalen und globalen Variablen ermöglicht, wie dies zum Beispiel bei den von den Autoren angeführten XPDL [27] oder BPEL der Fall ist.

Nach dem Prozess wird das zugehöriges Migrationsdatenmodell vom Prozessdesigner entworfen, das die Verteilung der Aktivitäten auf Prozessteilnehmer direkt vorgibt oder festlegt, wie diese bestimmt werden sollen. Sind der Entwurf von Prozess- und Migrationsdatenmodell abgeschlossen, werden diese beiden Dokumente auf eine Prozess-Engine deployed und zum Ausführen des Prozesses instanziiert, woraufhin die Ausführung des Prozesses beginnt (Abbildung 15).

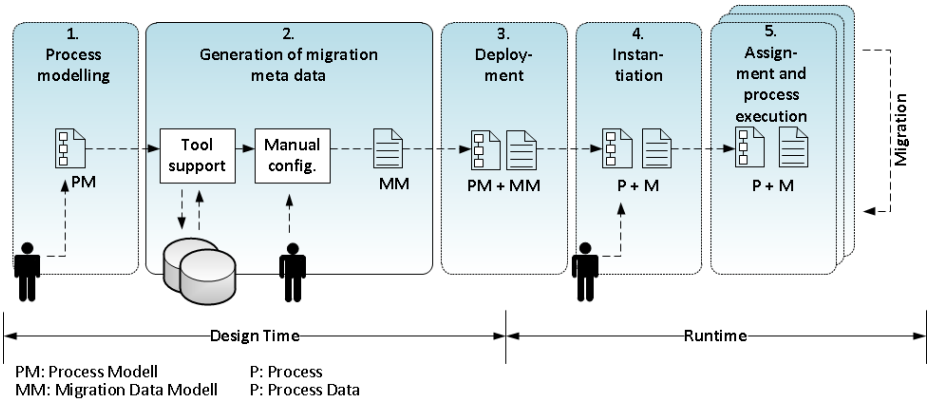

Abbildung 15: Methodik der Prozessmigration [1]

Wie bereits oben erwähnt erfolgt die Steuerung der Migrationen anhand der Migrationsdaten, die den Prozess um seinen Zustand und um Aktivitätszustände erweitern. In den Migrationsdaten ist es dazu für jede Aktivität möglich, entweder als Voreinstellung einen Teilnehmer oder eine Rolle festzulegen, Teilnehmer durch algorithmische Berechnungen bestimmen zu lassen oder anhand spezieller Aspekte der Dienstqualität bzw. des Kontextes festzulegen.

Die Migrationsdaten halten für jede Prozessinstanz und für jede Aktivität Zustände. So wird zu Beginn der Prozessausführung jede Aktivität auf *inactive* und der Prozesszustand auf den Zustand *Created* gesetzt. Befindet sich der Prozess in der Ausführung , was heißt, das gerade eine Aktivität ausgeführt wird, hat er den Zustand *Running*. Befindet sich ein Prozess im Zustand *Running*, kann er nicht migriert werden. Sind alle aktuellen Aktivitäten beendet und befinden sich im Zustand *executed*, wird der Prozesszustand auf *Option* gesetzt und eine Migration so ermöglicht. Eine Migration während der Ausführung einer Aktivität ist dadurch nicht möglich und verhindert somit Inkonsistenzen und bewahrt die Datenintegrität [1]. Für ein detaillierten Einblick in die verschiedenen Zustände des Prozesses oder der Aktivitäten wird an dieser Stelle auf [1] verwiesen.

Nachdem die Startaktivität in den Zustand *ready* gesetzt wurde, wird diese ausgeführt (Zustand *executing*) und wechselt nach Beendigung in den Zustand *executed*. Nun kann der Prozess zu einem anderen Teilnehmer migrieren oder

falls der aktuelle Teilnehmer die nächste Aktivität selbst ausführen kann, auf dem aktuellen Device verbleiben. Ist der aktuelle Teilnehmer nicht mehr in der Lage die nächste Aktivität auszuführen, wird der Prozess zu einem anderen Teilnehmer migriert.

Soll eine parallele Ausführung erfolgen, muss die Prozessinstanz repliziert werden, um eine Migration zu jedem Prozessteilnehmer der eine parallele Aktivität ausführt, zu ermöglichen. Hierzu wird von dem Prozessteilnehmer, der den ersten der parallelen Ausführungspfade ausführt, die erste Aktivität in den Zustand *executing* versetzt, der Prozess dann von der eigenen Prozessinstanz repliziert und dann an den Teilnehmer für den nächsten Pfad migriert. Hierdurch erkennt der nachfolgende Prozessteilnehmer, dass der erste Pfad bereits ausgeführt wird und er führt diesen nicht aus. Dies wird solange fortgesetzt, bis entweder alle parallelen Ausführungspfade abgearbeitet oder verworfen worden sind. Die Aktivität, die den parallelen Teilpfaden folgt, muss nun für eine korrekte Synchronisation und Zusammenführung der Teilpfade folgen. Wurden Variablen von mehreren Aktivitäten in verschiedenen Teilpfaden geschrieben, muss entschieden werden, welche der Zuweisungen im weiteren Verlauf des Prozesses verwendet werden soll. Dazu kommen in diesem Ansatz Datenklassen zum Einsatz. Datenklassen geben dabei beispielsweise an, ob die Daten während der parallelen Ausführung bereits synchronisiert werden müssen, ob die Daten nur ein gewisses Alter nicht überschreiten dürfen oder ob einfach der zuletzt geschriebene Wert der Variablen verwendet werden soll. Weiter Informationen zu Datentypen bietet hier [14].

2: $ADEPT_{distribution}$ [15] ist ein Workflow-Management-System, das Prozesse verteilt ausführen kann und auf ADEPT [17] basiert. Bei $ADEPT_{distribution}$ wird der ursprüngliche Prozess in sogenannte Partitionen logisch fragmentiert. Partitionen stellen eine Teilmenge der Aktivitäten des Prozesses dar und können statisch oder dynamisch zu Workflow-Servern zugewiesen werden [1]. Meistens erfolgt diese Zuweisung jedoch durch eine Kombination aus Vorberechnungen zur Designzeit und dynamischen Aspekten, wie beispielsweise Prozessumgebungsdaten. Die zugewiesenen Workflow-Server übernehmen die Ausführung und Kontrolle für die ihnen zugewiesenen Partitionen. In diesem Ansatz besteht das Workflow-Management-System aus verschiedenen Teilnetzen, die jeweils einen Workflow-Server besitzen. Die Benutzer des Systems sind einem oder mehreren der Teilnetze zugeordnet, jedoch müssen sie sich nicht in einem Teilnetz befinden, um als potentielle Bearbeiter derselben Aktivität zugewiesen zu werden.

Die Zuweisung der Aktivitäten zu Benutzern erfolgt in $ADEPT_{distribution}$ durch den Designer des Prozesses zur Entwurfszeit [13]. Für die Zuweisung werden jedoch üblicherweise Rollen oder Organisationseinheiten in Form von Prädikaten angegeben, und keine einzelnen Benutzer spezifiziert. Die Zuordnung der potentiellen Benutzer einer Aktivität erfolgt dann zur Laufzeit durch diese Auswahlprädikate. Welcher Benutzer die Aktivität dann ausführt, ist nicht

bestimmt, sondern ergibt sich zufällig, indem einer der potentiellen Benutzer die Aktivität auf seiner Arbeitsliste auswählt und ausführt. Unter einer Rolle versteht man in diesem Zusammenhang beispielsweise Berufe, wie Arzt oder Krankenschwester. Eine Organisationseinheit bezieht sich auf das Unternehmensumfeld in dem $ADEPT_{distribution}$ zum Einsatz kommt, was zum Beispiel in einem Krankenhaus die Chirurgie sein kann. Hierzu muss $ADEPT_{distribution}$ über ein Organisationsmodell verfügen, in dem Personen zu Organisationseinheiten und Rollen zugewiesen werden können.

Jeder Workflow-Server kontrolliert bei diesem Ansatz nur den Abschnitt des Prozesses, den seine Partition einschließt. Durch die verteilte Ausführung des Prozesses ist auch hier keine zentrale Serverinstanz notwendig, wodurch ein möglicher Flaschenhals während der Prozessausführung wegfällt. Da jedoch auch die Kommunikationsinfrastruktur bei großen Datenmengen problematisch werden kann, versucht $ADEPT_{distribution}$ benötigte Prozessmigrationen zu minimieren, um so die Menge der übertragenen Daten zu verringern. Zur Minimierung der benötigten Prozessmigrationen versucht der Ansatz, die Workflow-Server so zu wählen, dass möglichst nahe an den Benutzern bzw. Aktivitäten liegen [15]. Die zur bei der Vorberechnung der optimalen Workflow-Server berücksichtigten Kosten basieren auf den Transfers der Prozessdaten zwischen den einzelnen Aktivitäten, Aktualisierungen von Arbeitslisten, Kommunikationskosten mit externen Datenquellen (z.B Datenbanken) und den Migrationskosten für die Übergabe der Kontrolle vom aktuellen Workflow-Server an den Nächsten [17].

Eine Migration erfolgt in $ADEPT_{distribution}$ im Regelfall dann, wenn eine Kommunikation über Teilnetzgrenzen hinweg stattfinden muss. Das heißt, wenn der Benutzer der die aktuelle Aktivität ausführt, sich nicht im selben Teilnetz wie der Prozessteilnehmer, der die nachfolgende Aktivität ausführen soll, befindet, migriert die Prozessinstanz zum Workflow-Server des für die Folgeaktivität zuständigen Benutzers. $ADEPT_{distribution}$ setzt vorraus, dass sich die Organisationsstruktur und die Topologie der Kommunikationsinfrastruktur zur Ausführungszeit eines Prozesses nur minimal ändern dürfen.

3: Enabling Pervasive Execution of Workflows [18] beschreibt einen Ansatz zur verteilten Ausführung von Prozessen in pervasiven Umgebungen [28], also in Umgebungen, die bereits mit der heute verfügbaren Technik vollständig durchdrungen sind und somit dem mobilen Anwender vielschichtige Anwendungen bieten. Der Ansatz benötigt keinen zentralen Server zur Verwaltung des instanziierten Prozesses, da dies durch die Zusammenarbeit aller am Prozess beteiligten Device erledigt wird. Um diese verteilte Verwaltung des Prozesses zu ermöglichen, ist auf allen Devices eine Workflow-Engine implementiert, die im Wesentlichen eingehende Routingnachrichten empfängt, die vom Gerät auszuführenden Aktivitäten ausführt und Routingnachrichten zu nachfolgenden Devices schickt.

In dieser Arbeit wird der Workflow durch einen gerichteten Graphen repräsentiert. Die Knoten des Graphen stellen alle Aktivitäten dar, die von einem Device ohne Unterbrechung ausgeführt werden soll. Die Kanten des Graphen bilden den Ablauf des Prozesses ab. Für detaillierte Informationen wird an dieser Stelle auf [18] verwiesen. Eine feste Zuweisung von Benutzern oder Devices ist nicht vorgesehen, daher bietet der in der Arbeit vorgestellte Ansatz Rollen, anhand derer geeignete Devices ausgewählt werden können.

Die Autoren wählen für ihren Ansatz aufgrund der pervasiven Umgebung ein zustandsloses Protokoll, da eine Aktivität zu einem Zeitpunkt nur zu einem Device zugeordnet sein kann, und nach dem Migrieren der Prozessinstanz keine verbleibenden Informationen auf dem Device gespeichert werden. Um Prozesse zu migrieren wird ein web-service-basiertes Messaging-Protokoll vorgestellt, dessen Nachrichten eine *ID* zur Identifikation der Prozessinstanz, die Nummer der auszuführenden Aktivität, die komplette Workflow Definition, eine Abbildung zwischen Rollen und zugehörigen Devices und den Prozessdaten haben. Weiterführende Informationen finden sich in [18].

Wie bereits oben erwähnt wurde, hat jedes Device eine eigene Workflow-Engine. Die Engine stellt die Schnittstelle zwischen den vom Device unterstützten mobilen Anwendungen und den anderen Prozessteilnehmern dar. Die mobilen Anwendungen, die ein Device unterstützt, sind nicht nach außen sichtbar, sondern können nur durch die Workflow-Engine direkt gerufen werden, die als eine Art Proxy zwischen den Anwendungen und den anderen Devices des verteilten Prozesses agiert (Abbildung 16). Eine Aktivität ist in der Arbeit nicht als Einzelaktivität zu verstehen, sondern ist ein gekapselter interner Prozess, den das Device ausführt um eine Aufgabe zu erfüllen.

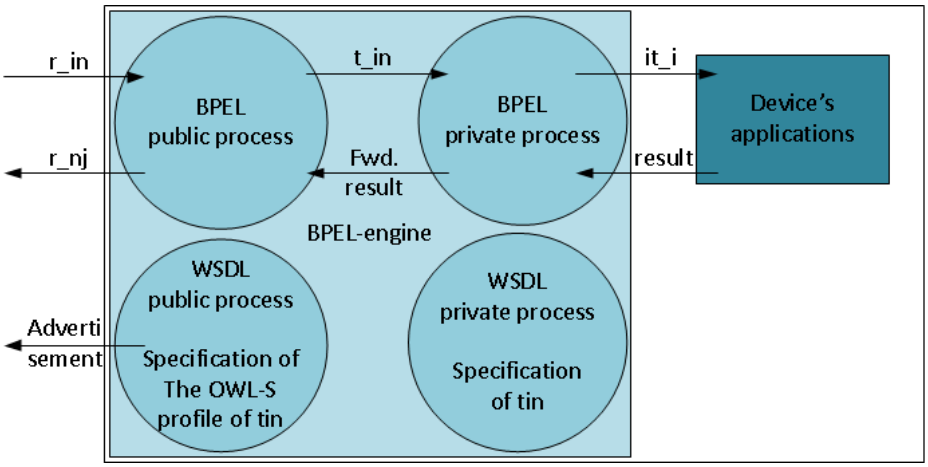

Abbildung 16: Infrastruktur der Devices Migration [18]

Zum Auffinden aller Devices bietet der vorgestellte Ansatz zwei Ansätze. Neben der Möglichkeit, dies während der Initialisierungsphase des Prozesses zu erledigen, erwägen die Autoren den Einsatz eines in der Arbeit nicht näher spezifizierten Dienstes, dem *Role Discovery Service*. Dieser Service soll von Devices, die die aktuelle Aktivität ausführen, parallel gerufen werden, um ein geeignetes Device als Nachfolger zu identifizieren.

4: OSIRIS [19] ist ein Workflow-Management-System zur verteilten Ausführung von Prozessen unter Verwendung einer Peer-to-Peer Infrastruktur. *OSIRIS* verfolgt dabei den Ansatz transaktionaler Prozesse [20], um eine korrekte Prozessausführung zu garantieren. Die Hauptkomponente in *OSIRIS* ist die sogenannte *Hyperbase-Datenbank*, die ein verteiltes Datenbank-Management-System ist. Die *Hyperbase-Datenbank* ist in allen am Prozess teilnehmenden Devices als *HDB-Layer* implementiert und übernimmt die Aufgabe Dienste anderer Devices zu verwenden und garantiert die Korrektheit der übertragenen Daten. Ein Client muss in *OSIRIS* also nicht selbst nach einem passenden Dienst suchen, da diese Routing-Aufgabe der *HDB-Layer* für ihn übernimmt. So muss das Device nur die Art des gewünschten Dienstes angeben. Die Prozessdaten werden in *OSIRIS* während der Ausführung einer Prozessinstanz als sogenanntes *Whiteboard* zwischen den Prozessteilnehmern migriert. Die Ausführung einer Prozessinstanz erfolgt in *OSIRIS*, wie in Abbildung 17 dargestellt wird, Peer-to-Peer-basiert.

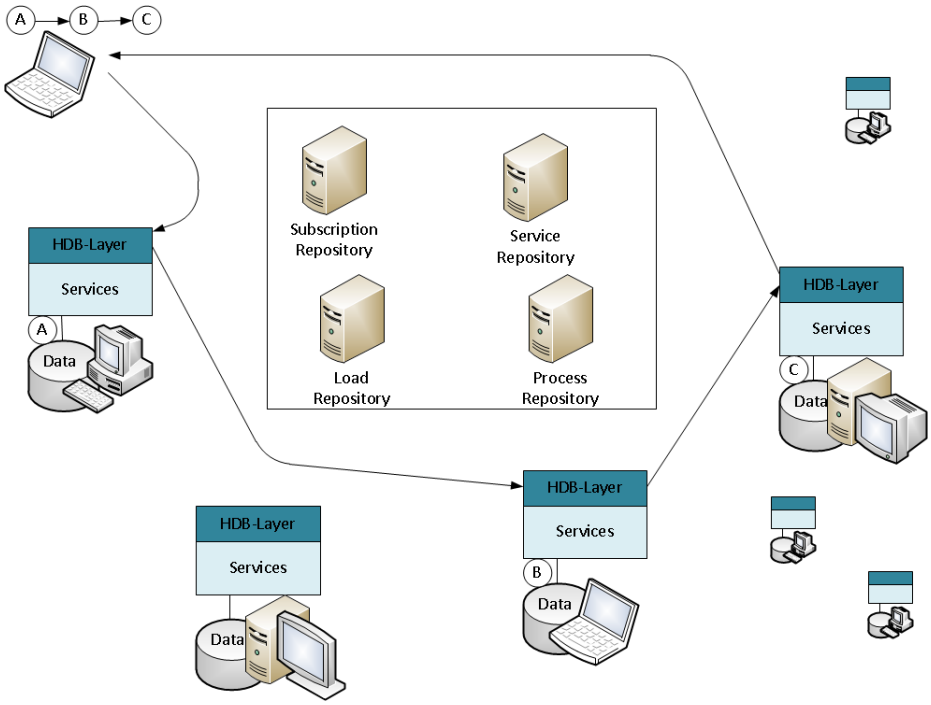

Abbildung 17: Peer-to-Peer-basierte Prozessausführung [19]

Um Anfragen der Devices durch die *HDB-Layer* zu bedienen, bietet der *OSIRIS*-Ansatz global verfügbare, aber verteilt gehaltene Verzeichnisse, wie das *Service Repository*, das *Subscription Repository*, das *Load Repository* und das *Process Repository*. Beispielsweise dient das *Service Repository* der Speicherung von Informationen, die die Dienste pro Device angeben und wird über ein „*publish & subscribe*"-Modell realisiert. Durch die verteilte Speicherung der Verzeichnisse wird in *OSIRIS* ein Engpass aufgrund zentraler Verzeichnisse vermieden. Um unnötige Replikationen der Verzeichnisse zu vermeiden, bietet *OSIRIS* auf Prädikaten basierende Mechanismen an, wie die partielle Replikation, bei der nur Daten an das teilnehmende Device übertragen werden, die diese Prozessinstanz betreffen. Es werden also nicht alle angebotenen Dienste von allen Devices in das *Service Repository* repliziert, sondern nur die von tatsächlich an der Ausführung beteiligten Devices.

Prozesse werden in *OSIRIS* als gerichtete Graphen dargestellt, wobei die Knoten die Aktivitäten darstellen und die Kanten die Ordnung der einzelnen Aktivitäten abbilden. Ein Prozess verfügt über exakt eine Start- und eine Endaktivität. Eine Aktivität kann nur dann ausgeführt werden, wenn alle Vorgängeraktivitäten beendet wurden und alle Bedingungen für die Ausführung erfüllt sind. Aktivitäten können in *OSIRIS* kompensierbar, wiederholbar oder sogenannte *pivot*-Aktivitäten sein. Kompensierbare Aktivitäten können nach ihrer Ausführung wieder zurückgesetzt werden. Wiederholbare Aktivitäten können bei fehlerhafter Ausführung so lange erneut ausgeführt werden, bis sie erfolgreich beendet worden sind. *Pivot*-Aktivitäten sind nicht kompensierbar und so ist nach dem Abschluss einer solchen Aktivität kein Rücksetzen mehr möglich.

Die Zuweisung von Aktivitäten zu Devices erfolgt in *OSIRIS* nicht zur Entwurfszeit durch den Designer des Prozesses, um ein LOAD-Balancing durch die Verwendung des *Load Repository* zu ermöglichen. Der Designer spezifiziert hier nur den Typ, den der Dienst zur Ausführung einer Aktivität benötigt, da konkrete Zuweisungen zur Laufzeit erfolgen.

Sollen Aktivitäten parallel ausgeführt werden, so geschieht dies durch den Aufruf paralleler Ausführungszweige im Prozessgraphen. Jeder der parallelen Zweige erhält dabei nur den Teil des *Whiteboards*, den für die Ausführung seiner Aktivitäten benötigt wird [21]. Eine Join-Aktivität führt die parallelen Ausführungszweige nach deren Beendigung zusammen und setzt die Teile des *Whiteboards* wieder zusammen.

5: Exploring Transactional Service Properties for Mobile Service Composition [22] befasst sich mit den transaktionsbezogenen Eigenschaften von Prozessen mit dem Fokus auf mobile Umgebungen. Der Ansatz der Arbeit ist eine transaktionsbezogene Koordination und Zusammensetzung von Diensten, also Prozessen, in mobilen Netzwerken zu ermöglichen. Obwohl in dieser Arbeit kein direkter Bezug zur logischen Fragmentierung von Prozessen gegeben ist, wurde dieser Ansatz dennoch bei den Ansätzen der logischen Fragmentierung mit aufge-

nommen. In der Arbeit setzen sich Prozesse aus Transaktionen zusammen, die die Serviceaufrufe während der Prozessausführung koordinieren. Durch diese transaktionsbasierte Koordination der Prozessaktivitäten erfolgt ebenfalls eine logische Fragmenierung des vollständigen Prozesses.

Das zugrundeliegende Modell der Dienste ist zustandsbasiert. Zu Beginn hat ein Dienst den Zustand *initial*, was bedeutet, dass die Dienst noch nicht aktiviert wurde. Nachdem der Dienst aktiviert wurde, befindet er sich im Zustand *activated* und kann in die Zustände *failed* oder *cancelled* übergehen, wenn beispielsweise interne oder externe Fehler auftreten. War die Ausführung erfolgreich, wechselt der Dienst in der Zustand *completed*. Zusätzlich besteht die Möglichkeit, dass kompensierbare, dass heißt zurücksetzbare Dienste in den Zustand *compensated* wechseln können. Hierbei kann ein Wechsel in die Zustände *actived*, *cancelled* und *compensated* nur von außen, also zum Beispiel durch die Workflow-Engine eingeleitet werden, die anderen Zustände können durch den Dienst selbst angesteuert werden.

Die Routine zur standardmäßigen Fehlerbehandlung ist das Anhalten von laufenden Diensten, das Kompensieren von abgeschlossenen Diensten und die Aktivierung alternativer Ausführungen. Um diesen Mechanismus abzubilden, geben die Autoren die daraus resultierenden Abhängigkeiten, in denen Dienste untereinander stehen können, an. Für detaillierte Informationen wird hier auf die Arbeit verwiesen.

Ein Prozess besteht aus einer Menge von Diensten und den Beziehungen, in denen sie untereinander stehen. Durch die transaktionsbezogene Zusammensetzung der Prozesse, können zusätzlich zu den Beziehungen, in denen sie im Normalfall, also ohne auftretende Fehler ausgeführt werden, noch die Beziehungen der Dienste beim Auftreten von Fehlern angegeben werden, um inkonsistente Prozesszustände zu vermeiden. Die Struktur der Prozesse wird durch Workflow-Muster, wie Sequenzen, die parallele Ausführung oder das exklusive Oder abgebildet. Durch diese Muster wird der Kontrollfluss und die Ausführungssemantik der Prozesse abgebildet.

Da die Autoren die Blockade von Ressourcen vor allem im Umfeld mobiler Prozesse für kontraproduktiv halten, führen sie ein abgeschwächtes Korrektheitskriterium, die *Semi-Atomarität* bei transaktionalen Prozessen ein. Unter *Semi-Atomarität* verstehen die Autoren, dass vor der Bindung von nicht kompensierbaren, also *Pivot*-Diensten, erst kompensierbare Dienste gebunden werden. Für eine formale Definition der *Semi-Atomarität* wird an dieser Stelle auf die Arbeit verwiesen. Zur Einhaltung des Kriteriums geben die Autoren zusätzlich Mechanismen zur Umordnung der Dienste an.

Zwischenfazit Nach der detaillierten Betrachtung verschiedener Ansätze zur logischen Fragmentierung von Prozessen mit dem Ziel einer verteilten Ausfüh-

rung der Prozesse fällt auf, dass alle Ansätze die ursprüngliche Struktur des Prozesses erhalten und die Prozessinstanz unter allen teilnehmenden Devices migriert. Dadurch erlangen alle, am Prozess beteiligten Devices, Kenntnis vom vollständigen Prozess, weshalb eine gute Fehlerbehandlung möglich wird.

Die Allokation der Devices oder Benutzer erfolgt bei allen Ansätzen entweder vollkommen dynamisch, zum Beispiel durch algorithmische Berechnungen oder kann zumindest durch Rollen festgelegt werden, wodurch die Ansätze die notwendige Flexibilität bieten, die für die Ausführung mobiler Prozesse notwendig ist. Benutzerwechsel sind während der Ausführung einer atomaren Aktivität nicht vorgesehen, können jedoch sonst zu jedem Zeitpunkt im Prozess erfolgen, was die Flexibilität weiter steigert.

Wird für die Migrationsdaten ein zusätzliches Dokument verwendet, ist überdies sogar die Kompatibilität mit herkömmlichen, zentral gesteuerten Workflow-Management-Systemen möglich. Durch ein Verwerfen der Migrationsdaten ist in diesen Systemen eine Ausführbarkeit der Prozesse weiterhin gegeben.

Die logische Fragmentierung bietet somit eine sehr große Flexibilität. Da auch die Fehlerbehandlung von den meisten Ansätzen berücksichtigt wird, eignet sich die logische Fragmentierung bestens für den Einsatz im Umfeld verteilter mobiler Prozesse. Im folgenden werden die unterschiedlichen Ansätze der physischen und logischen Fragmentierung diskutiert und verglichen.

5.3 Diskussion

In diesem Abschnitt werden die bisher betrachteten Ansätze der verteilten Prozessausführung auf ihre Erfüllung der einzelnen Anforderungen untersucht. Geprüft werden die Ansätze auf die Erfüllung der folgenden Kriterien, die sich aus der vorhergegangenen Literaturbetrachtung ergeben:

Allgemeine Kriterien:

1.1 Ausführung ohne zentralen Server (Kapitel 5 - Anforderungen mobiler Prozesse an ein Workflow-Management-System / Kontrollfluss)

1.2 Dezentrale Datenspeicherung (Kapitel 5 - Anforderungen mobiler Prozesse an ein Workflow-Management-System / Datenfluss)

2. Unterscheidung globaler sowie lokaler Variablen (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Daten): Lokale Variablen werden nur innerhalb von Aktivitäten oder Teilprozessen verwendet und kommen hauptsächlich steuernd zum Einsatz. Sie sind für andere Prozessteilnehmer nicht sichtbar und auch nicht für das Ergebnis des Prozesses wichtig. Globale Variablen sind im Gegensatz zu lokalen Variablen für alle Prozessteilnehmer sichtbar.

3.1 Flexible Zuweisung von Devices (Kapitel 4.3 - Herausforderungen mobiler Prozesse / Allokation): Devices können nicht ausschließlich statisch zur Designzeit oder kurz nach der Instanziierung allokiert werden, sondern können auch dynamisch während der Prozessausführung zugewiesen werden.

3.2 Benutzerwechsel möglich (Kapitel 5 - Anforderungen mobiler Prozesse an ein Workflow-Management-System / Benutzer / Devices)

3.3 Devicewechsel möglich (Kapitel 5 - Anforderungen mobiler Prozesse an ein Workflow-Management-System / Benutzer / Devices)

4. Prozessstrukur bleibt erhalten (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Prozessstruktur): Bei der physischen Fragmentierung von Prozessen wird der Prozess aufgebrochen und physisch verteilt, wodurch die Prozessstruktur nicht erhalten bleibt. Bei logischen Fragmentierungsansätzen bleibt die Prozessstruktur in der Regel erhalten oder wird nur geringfügig abgeändert.

5. Mobile Workflow-Engine kann Prozessfragmente ausführen (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Mächtigkeit der mobilen Engine): Je nach Mächtigkeit der mobilen Engine, kann diese nur einzelne Aktivitäten ausführen oder auch größere Prozessfragmente, die aus mehreren Aktivitäten bestehen, selbstständig ausführen.

6. Konzept orientiert sich an Infrastruktur (4.1 Mobile Prozesse): Die Abbildung des Geschäftsprozesses, dessen Fragmentierung und die Ausführung orientieren sich an der gegebenen Infrastruktur.

Partitionierung	Ansatz	Merkmale								
		1		2	3			4	5	6
		1	2		1	2	3			
physisch	CiAN	J	J	-	N	N	N	N	N	J
	MobiWork	J	J	-	N	N	N	N	N	J
	Mentor	J	J	-	J	N	N	N	N	J
	A Novel Approach to dezentralized Workflow Enactement	J	J	J	N	-	-	N	N	J
logisch	Flexible Execution of Distributed Business Processes based on Process Instance Migration	J	J	J	J	-	-	J	J	J
	Adept distribution	J	J	-	J	J	J	-	-	J
	Enabling Pervasive Execution of Workflows	J	J	J	J	-	-	J	J	J
	OSIRIS	J	J	-	J	-	-	J	-	J
	Exploring Transactional Service Properties for Mobile Service Composition	J	J	-	-	-	-	-	-	J

Tabelle 3: Vergleich aller untersuchten Ansätze (Allgemeine Kriterien)

Migrationsbezogene Kriterien:

1. Migrationsentscheidung zur Designzeit (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Migrationsentscheidung): Migrationen werden zur Designzeit festgelegt, das heißt bereits zur Designzeit muss bekannt sein, wo welche Aktivität ausgeführt werden soll.

2. Migrationsentscheidung zur Laufzeit (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Migrationsentscheidung): Migrationen werden dynamisch zur Laufzeit des Prozesses festgelegt.

3.1 Migrationsentscheidung durch Regeln (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Migrationsentscheidung): Die dynamische Migrationsentscheidung wird aufgrund von Regeln getroffen.

3.2 Migrationsentscheidung durch Rollen-/Benutzerzuweisungen (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Migrationsentscheidung): Die Migrationsentscheidung wird aufgrund von Rollen-/Benutzerzuweisungen getroffen.

3.3 Migrationsentscheidung durch Berechnungen/Routing (Kapitel 5.2 Logische Fragmentierung / Klassifikationsmerkmale / Migrationsentscheidung): Die dynamische Migrationsentscheidung wird aufgrund von Berechnungen aus den der Prozessausführungsumgebung getroffen.

Ansatz	Merkmale				
	1	2	3		
			1	2	3
Flexible Execution of Distributed Business Processes based on Process Instance Migration	J	J	J	J	J
Adept distribution	-	J	J	J	J
Enabling Pervasive Execution of Workflows	N	J	-	J	-
OSIRIS	N	J	N	N	J
Exploring Transactional Service Properties for Mobile Service Composition	-	-	-	-	-

Tabelle 4: Vergleich der untersuchten logischen Ansätze (Migrationsbezogene Kriterien)

Tabelle 3 und Tabelle 4 vergleichen die in dieser Arbeit betrachteten Ansätze der verteilten Ausführung von Prozessen. Ist bei einem Kriterium ein „-“ eingetragen, bedeutet dies, dass die Arbeiten zu den Ansätzen die Thematik gar nicht behandelt haben oder aus Vereinfachungsgründen von einer Betrachtung abgesehen haben.

Wie man an Tabelle 3 leicht sehen kann, erfüllen die ersten vier Ansätze die meisten Kriterien, was die Flexibilität und Dynamik von Zuordnungen von Benutzern oder Devices angeht nicht. Dies liegt darin begründet, dass diese

Ansätze eine physische Fragmentierung vornehmen, die entweder zur Designzeit oder kurz nach der Initialisierung des Prozesses die Devices zu den Aktivitäten zuordnet. Diese starre Zuordnung und die physische Fragmentierung der originalen Prozesse führen dazu, dass diese Ansätze sich nur bedingt eignen, um flexibel auf Änderungen in mobilen Prozessen reagieren zu können.

Die fünf anderen Ansätze sind Ansätze, die eine logische Fragmentierung der Prozesse anwenden und die Prozessinstanz zu den der aktuellen Aktivitäten / Fragmente zugeordneten Workflow-Engines migrieren. Durch die Migration der Prozessinstanzen ist bei allen Ansätzen eine Prozesskenntnis der Prozessteilnehmer gegeben, wodurch auftretende Fehler leichter durch Behandlungsroutinen behoben werden können (mittels abgestimmter Koordination). Eine dynamische Zuordnung der Prozessteilnehmer ist in allen Ansätzen möglich. Leider gehen auch hier die meisten Ansätze nur unzureichend auf die Möglichkeit ein, dass während der Ausführung eines Fragments der ausführende Benutzer oder das Device wechseln können (Tabelle 4). Da der letzte Ansatz sich im Wesentlichen mit den Möglichkeiten mobile Dienste transaktional zu koppeln beschäftigt, geht dieser kaum auf Details der verteilten Ausführung, Device- oder Benutzerzuordnungen ein.

Allen Ansätzen gemein ist, dass sie sich stark eingesetzten Infrastruktur orientieren, wodurch jedoch Einschränkungen beim Design der Prozesse auftreten können und der Designer eventuell nicht alle erforderlichen Geschäftsprozesse abbilden kann.

Der Ansatz „Flexible Execution of Distributed Business Processes based on Process Instance Migration ist aus Sicht dieser Arbeit am besten für den Einsatz im Umfeld mobiler Prozesse geeignet, da er für die Zuordnung der Devices und / oder User die größte Flexibilität aufweist.

Die bisher betrachteten Ansätze bieten für den Einsatz in Business-Process-Management-Systemen große Potenziale. Im folgenden Kapitel wird anhand eines kommerziellen Beispielproduktes, dem Microsoft BizTalk Server, das aufgrund seines großen Funktionsumfangs ausgewählt wurde, die Umsetzung eines mobilen Prozesses detailliert beschrieben. Zusätzlich bietet das folgende Kapitel einen knappen Einblick in die Architektur des BizTalk Servers.

6 BizTalk Server

Der Microsoft BizTalk Server [29] (Abbildung 18) ist ein kommerzielles System, das die Funktion eines zentralen Message Broker übernimmt, der in den Bereichen Enterprise Application integration (EAI), Business-to-Business (B2B), Serviceorientierte Architekturen (SOA) sowie Business Process Management (BPM) zum Einsatz kommen kann. Auf den Seiten des Herstellers wird zudem

der Einsatz mobiler Technologie durch den Aufsatz des Microsoft BizTalk RFID Servers beschrieben.

Abbildung 18: BizTalk Logo

Abbildung 7 zeigt die Architektur des BizTalk Servers. Das zentrale Element des BizTalk Servers ist die Message-Box, eine SQL-Datenbank, die den Nachrichtenaustausch zwischen allen an einem Prozess beteiligten Systemen steuert. Die Message-Box nimmt hierzu Nachrichten entgegen und leitet diese an Zielsysteme weiter, die eine Subscription für diese Nachrichtenart haben.

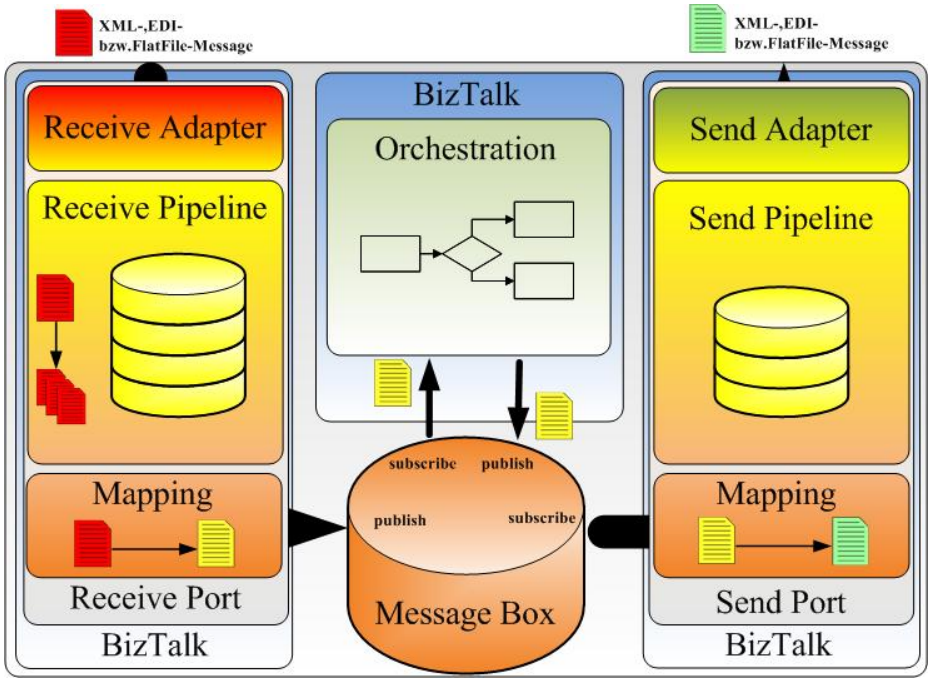

Abbildung 19: BizTalk Engine

Nachrichten können entweder direkt von der Massage-Box an entsprechende Zielsysteme weitergeleitet werden oder zuvor von der Orchestration-Engine verarbeitet werden, die die eigentliche Prozesssteuerung übernimmt. Für den Empfang der Nachrichten stehen dem Entwickler des BizTalk Servers die Receive und Send Ports zur Verfügung, deren einzelne Komponenten die Umwandlung der Nachrichten in das vom Zielsystem benötigte Format und ein eventuell erforderliches Mapping der einzelnen Quellfelder auf Zielfelder realisieren.

Um nun einen Prozess mit dem BizTalk Server steuern zu können, ist es zunächst notwendig, alle Systeme mit die am Prozess teilnehmen sollen an den BizTalk anzubinden. Hierzu müssen in einem ersten Schritt für jedes System entsprechende Send- und Receive-Ports erstellt werden. In den Receive-Ports werden die eingehenden Nachrichten über die Receive Pipeline in ein XML-basiertes Format gebracht. Dazu ist es nötig das Format für jeden eingehenden Nachrichtentyp vorab im XML-Format zu definieren. Um die Modellierung des Prozesses zu vereinfachen, kann im BizTalk zusätzlich ein eigenes internes XML-basiertes Nachrichtenformat definiert werden, so dass die Orchestrierung nur mit einem Nachrichtenformat umgehen muss. Wurde ein solches Format definiert, werden die nach der Receive Pipeline bereits in XML vorliegenden Nachrichten vom Mapper in das interne Nachrichtenformat transformiert. Send-Ports sind prinzipiell Umkehrungen der Receive-Ports. Für detailliertere Informationen über die Send- und Receive-Ports wird an dieser Stelle auf [23] verwiesen.

Möchte man nun einen mobilen Prozess mit dem BizTalk Server verwalten, und setzt nur „relativ einfache“mobile Zusatzgeräte wie RFID-Leser ein, so ist dies durch die guten Verwaltungsmöglichkeiten der Geräte im BizTalk RFID Server gut realisierbar. Da der BizTalk Server selbst allerdings keine fertige Engine für mobile Devices bereitstellt, muss der Anwender diese selbst implementieren. Dazu stellt der BizTalk Server Schnittstellen bereit, die Implementierungen in der Programmiersprache C# oder anderer .Net-Sprachen voraussetzen.

Abschließend kann festgestellt werden, dass der BizTalk zur Verwaltung von Prozessen einen hohen Aufwand für die Definition der Nachrichtenformate und der Kommunikationsports erfordert. Sollen zudem mobile Prozesse im Sinne dieser Arbeit umgesetzt werden, muss ein hohen Implementierungsaufwand betrieben werden. Der BizTalk Server ist also eher als ESB [34] einsetzbar.

7 Fazit und Ausblick

Nach Betrachtung und Vergleich der unterschiedlichen Ansätze der verteilten Ausführung von Prozessen, können Ansätze, die statische Zuweisungen von Bearbeitern zu Aktivitäten erfordern, als zu unflexibel für mobile Prozesse angesehen werden. Die physische Fragmentierung ist somit zu starr, um ausreichend auf die Anforderungen im Umfeld mobiler Prozesse eingehen zu können.

Im Gegensatz zur physischen Fragmentierung bietet die logische Fragmentierung durch den Einsatz von Prozessmigrationen und variablen Zuweisungen von Benutzern, Devices und verteilten Servern die größtmögliche Flexibilität, um die Anforderungen mobiler Prozesse zu erfüllen. Um der hohen Dynamik und den großen Unsicherheiten in mobilen Umgebungen Rechnung zu

tragen, wird daher der Einsatz einer auf logischer Fragmentierung basierenden Lösung empfohlen.

Auf dem Markt für Business-Process-Management existiert eine Vielzahl an kommerziellen Produkten mit unterschiedlichem Funktionsumfang wie IBM's Websphere [37], Oracle BPM [38] oder SAP Netweaver Business Process Management [39]. Wie man am im Rahmen dieser Arbeit betrachteten Beispielprodukt Microsoft BizTalk Servers sehen kann, unterstützt dieses kommerzielle System den Anwender noch nicht ausreichend bei der Erstellung und Verwaltung mobiler Prozesse sowie der teilnehmenden Devices.

Zukünftig müssen die Hersteller von Business-Process-Management-Systemen und Workflow-Management-Systemen weiter daran arbeiten, eine gute Unterstützung für den Einsatz mobiler Technologien zu bieten. Aufgrund der immer größer werdenden Mobilität unserer Gesellschaft und dem ungebrochenen Trend an jedem Ort und zu jeder Zeit seine Daten, wie E-Mails, oder Dienste in Anspruch zu nehmen, sind die Hersteller angehalten noch mehr Arbeit in diesem Bereich zu zeigen. Des Weiteren müssen die Hersteller eine noch größere Flexibilität der eingesetzten Devices und der gesamten Infrastruktur erreichen, um zukünftig mobile Prozesse unabhängig von der eingesetzten Technologie, also ausschließlich abhängig von dem abzubildenden Geschäftsprozess, modellieren zu können.

Durch den vermehrten Einsatz mobiler Technologien werden sich in vielen Bereichen große zusätzliche oder neue Potentiale erschließen. Wie bereits eingehend erwähnt, kann durch den Einsatz verteilter mobiler Technologie im Gesundheitswesen, beispielsweise bei Visiten in Krankenhäusern Zeit gespart werden, da nach der Visite des Patienten für die neue Einstellung der Medikamentierung keine zusätzlichen Absprachen oder Anweisungen mehr nötig sind. Auch die begleitende Betreuung von Erkrankten kann durch den unterstützenden Einsatz von mobilen Devices erheblich erleichtert werden.

Literatur

1. Sonja Zaplata, Kristof Hamann, Kristian Kottke, Winfried Lamersdorf: Flexible Execution of Distributed Business Processes based on Process Instance Migration, Journal of Systems Integration, Vol 1, No 3, 2010
2. Stefan Jablonski, Ralf Schamburger, Cristian Hahn, Stefan Horn, Rainer Lay, Jens Neeb, Michael Schlundt: A Comprehensive Investigation of Distribution in the Context of WorkflowManagment, Proceedings. 8th Internal Conference on Parallel and Distributed Systems, 2001
3. Tobias Binz, Jan Königsberger, Tina Schliemann: Vergleich von Ansätzen zur Fragmentierung von Prozessen, Fachstudie, Universität Stuttgart, 2009
4. Ute Bachmeyer: Integriertes Daten- und Prozessmanagement in mobilen Umgebungen am Beispiel einer mobilen Außendienstanbindung innerhalb des Liebherr-Konzerns, Diplomarbeit, Universität Ulm, 2007
5. Andre Köhler, Volker Gruhn: Lösungsansätze fur verteilte mobile Geschäftsprozesse, Elektronische Geschäftsprozesse, 2004
6. Rohan Sen, Gruita-Catalin Roman, Christopher Gill, Andrew Frank: CiAN: A Workflow Engine for MANETs, Lecture Notes in Computer Science Volume 5052, 2008
7. Gregory Hackmann, Rohan Sen, Mart Haitjema, Gruia Catalin Roman, Christopher Gill: MobiWork: Mobile Workflow for MANETs, Wissenschaftliche Arbeit, University Washington, 2006
8. Peter Muth, Dirk Wutke, Janine Weißenfels,Angelika Kotz Dittrich, Gerhard Weikum: From Centralized Workflow Specification to Distributed Workflow Execution, Journal of Intelligent Information Systems Volume 10 Issue 2, 1998
9. Dirk Maier: Gossiping, Diplomarbeit, Eidgenössische Technische Hochschule Zürich, 2004
10. Daniel Martin, Daniel Wutke, Frank Leymann: A Novel Approach to Decentralized Workflow Enactment, 12th International IEEE Enterprise Distributed Object Computing Conference, 2008
11. Michael Scherrer: Linda und Tupel-Spaces, Seminararbeit, Universität Würzburg, 2005
12. http://www.microsoft.com/germany/biztalk/einsatz/default.mspx, zuletzt besucht: 16.11.2010
13. Thomas Bauer, Manfred Reichert, Peter Dadam: Effiziente Durchführung von Prozessmigrationen in verteilten Workflow-Management-Systemen, Wissenschaftliche Arbeit, Universität Ulm, 2000
14. Kristof Hamann: Parallele Ausführung von Prozessen auf mobilen Geräten, Diplomarbeit, Universität Hamburg, 2009
15. Thomas Bauer, Peter Dadam: Efficient Distributed Workflow Management Based on Variable Server Assignment, In Advanced Information Systems Engineering, 12th International Conference CAiSE 2000 Vol. 1789, 2000
16. Sonja Zaplata, Winfried Lamersdorf, Christian Kunze: Kontextbasierte Kooperation fur mobile Geschäftsanwendungen Dezentrale Ausfuhrung und Management von mobilen Prozessen, WIRTSCHAFTSINFORMATIK Ausgabe Nr.: 2009-04, 2009
17. Jochen Zeitler : Integration von Verteilungskonzepten in ein adaptives Workflow-Management-System, Diplomarbeit, Universität Ulm, 1999
18. Frederic Montagut, Refik Molva: Enabling Pervasive Execution of Workflows, International Conference on Collaborative Computing: Networking, Applications and Worksharing, 2005

19. Christoph Schuler, Roger Weber, Heiko Schuldt, Hans-J. Schek: Scalable Peer-to-Peer Management - The OSIRIS Approach, Proceedings of the 2 nd International Conference on Web Services, 2004
20. http://www.openworkflow.org/websites/openworkflow.nsf/chapter/0030.0045.0030.,zuletzt besucht 17.11.2010
21. Daniel Wutke: Eine Infrastruktur fur die dezentrale Ausfuhrung von BPEL-Prozessen, Dissertation, Universtitä Stuttgart, 2010
22. Katharina Hahn, Heinz F. Schweppe: Exploring Transactional Service Properties for Mobile Service Composition, Proceedings zur 4. Konferenz Mobile und Ubiquitäre Informationssysteme, März 2009
23. Jörg Grüning: Microsoft BizTalk Server, Seminararbeit, Universität Ulm, 2010
24. Manfred Hauswirth, Schahram Dustdar: Peer-to-Peer: Grundlagen und Architektur, Datenbank Spektrum Heft 13, 2005
25. IEEE Standard for Information technology Telecommunications and information exchange between systems Local and metropolitan area networks Specific requirements, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 2007
26. Web Services Business Process Execution Language Version 2.0, OASIS Standard, 2007
27. Workflow Management Coalition: Process Definition Interface – XML Process Definition Language, Workflow Standard, 2008
28. Anthony LaMarca, Marc Langheinrich, Khai Truong: Pervasive Computing, Proceedings of the 5th International Conference, PERVASIVE, 2007
29. Microsoft Corporation: Microsoft BizTalk Server 2010 Technical Overview, 2010
30. Peter Dadam, Manfred Reichert: The ADEPT project: a decade of research and development for robust and flexible process support - Challenges and Achievements, Computer Science - Research and Development Volume 23, 2009
31. Manfred Reichert, Stefanie Rinderle-Ma, Peter Dadam: Flexibility in Process-Aware Information Systems, TRANSACTIONS ON PETRI NETS AND OTHER MODELS OF CONCURRENCY II, 2009
32. Manfred Reichert, Thomas Bauer, Peter Dadam: Flexibility for Distributed Workflows, Handbook of Research on Complex Dynamic Process Management: Techniques for Adaptability in Turbulent Environments, 2009
33. Manfred Reichert: Dynamische Ablaufänderungen in Workflow-Management-Systemen, Dissertation, Universität Ulm, 2000
34. Rüdiger Pryss: Enterprise Application Integration: Anforderungen, Ansätze und Technologien, Diplomarbeit, Universität Ulm, 2005
35. Peter Dadam, Manfred Reichert, Stefanie Rinderle, Hilmar Acker, Martin Jurisch, Ulrich Kreher,Kevin Göser, Markus Lauer: ADEPT2 - Next Generation Process Management Technology, Proceedings Fourth Heidelberg Innovation Forum (Invited Paper), 2007
36. Gustavo Alonso, Fabio Casati, Harumi Kuno, Vijay Machiraju: Web Services, Springer, 2003
37. http://www-01.ibm.com/software/de/websphere/, zuletzt besucht 20. November 2010
38. http://www.oracle.com/us/technologies/bpm/index.html, zuletzt besucht 20. November 2010
39. http://www.sap.com/germany/plattform/netweaver/components/sapnetweaverbpm/index.epx, zuletzt besucht 20. November 2010
40. Christian P. Kunze, Sonja Zaplata, Winfried Lamersdorf: Mobile Process Description and Execution, Lecture Notes in Computer Science, Volume 4025, 2006

Abbildungen

Tabellen

JÖRG GRÜNING

Eidesstattliche Erklärung

Hiermit erkläre ich, dass ich diese Arbeit und nur unter Zuhilfenahme der angegebenen Quellen und Hilfsmittel selbstständig verfasst habe.

Diese Arbeit wurde bisher weder einer anderen Prüfungsbehörde vorgelegt, noch anderweitig in irgendeiner Form veröffentlicht.

Ulm, den 22. November 2010

Jörg Grüning (Mat.-Nr: 642588)