Unterstützung mobiler Prozesse im Mobile Computing

Christian P. Kunze
Verteilte Systeme und Informationssysteme
Fachbereich Informatik, Universität Hamburg
Vogt-Kölln-Str. 30, D-22527 Hamburg
kunze@informatik.uni-hamburg.de

1 Einleitung

Mobilität ist immer noch einer der wichtigsten Teilbereiche im *Pervasive Computing*. Die Überwindung der sich aus ihr ergebenden Anforderungen und Einschränkungen ist dabei eine der Aufgaben von Middleware-Systemen des *Mobile Computing*. Es hat sich gezeigt, dass solche Systeme im Gegensatz zu klassischen verteilten Systemen, meist weniger (Verteilungs-) Transparenz, dafür aber mehr Ortsbezug sowie ein Bewusstsein über ihre Mobilität (*Awareness)*, besitzen sollten. Ausgehend von diesem Wissen ist es dann möglich, Anwendungen an die Bedingungen der sich verändernde Umgebung anzupassen (*Adaptability*).

Diese Awareness und Adaptability beschränkt sich bei aktuellen Middleware-Ansätzen zur Unterstützung mobiler Anwendungen derzeit in den meisten Fällen jedoch darauf, dass mehr oder weniger monolithisch strukturierte Anwendungen in der Ausführung einzelner, kurzzeitiger Aufgaben unterstützt werden. Um jedoch der Vision des Pervasive Computings näher zu kommen, sollten Middleware-Systeme in diesem Bereich auch komplexere, möglichst sogar a priori unbekannte und vor allem langlebige Aufgaben unterstützen können.

Solche komplexen und anwendungsbezogenen Aufgaben können als eine Sequenz von in Beziehung stehenden einfachen Diensten angesehen werden, die in einem „Prozess“ zusammengefasst werden. Solche Prozesse werden dann (u.a.) von mobilen Clients im Interesse seines Benutzers verwaltet und ausgeführt. Dies führt jedoch zu der Konsequenz, dass der mobile Client nun alle Dienste des Prozesses kennen und ausführen können muss. Zudem muss er in der Lage sein, alle Zwischenergebnisse verarbeiten zu können

– egal welche Größe oder Relevanz sie für das gewünschte Ergebnis haben. Die Konsequenz hieraus ist, dass nun die mobilen Geräte zum Flaschenhals werden und ihre begrenzten Fähigkeiten die Menge der möglichen Prozesse bestimmen.

Weil die meisten Benutzer und Benutzerinnen jedoch lediglich am Endergebnis bzw. an einigen spezifischen Effekten des Prozesses interessiert sind (und nicht an irgendwelchen Zwischenergebnissen oder der Art der Ausführung selbst), besteht in Verteilten Umgebungen auch die Möglichkeit, den Kontrollfluss – und mit ihm den ganzen Prozess – an andere Ausführungseinheiten zu delegieren. In Kombination mit der Möglichkeit von Middleware-Systemen für das Mobile Computing, Kontextinformationen zu nutzen und untereinander zu kooperieren, kann damit die Integration von langlebigen mobilen Prozessen und deren verteilte Ausführung einen zusätzlichen Nutzen auch im Bereich des Mobile Computings bringen.

Im weiteren Verlauf dieses Beitrags wird das Projekt *Distributed Environment for Mobility-Aware Computing* (DEMAC) mit seinen Konzepten und Realisierungsvorschlägen zur Unterstützung mobiler Prozesse im Mobile Computing vorgestellt (Abschnitt 3). Zuvor werden in Abschnitt 2 die im Projekt betrachteten bestehenden Infrastrukturen und Prozessmodelle beschrieben. Abschnitt 4 beschließt den Beitrag mit einer kurzen Zusammenfassung und einem Ausblick.

2 Bestehende Infrastrukturen und Prozessmodelle

Mit dem Ziel, langlebige benutzer-zentrische Prozesse in eine Middleware für mobile Systeme zu integrieren, müssen Arbeiten zum einen im Bereich des Mobile Computings und zum anderen im Gebiet der Prozessbeschreibung und -ausführung betrachtet werden. Bei der Analyse der Middleware-Systeme soll ergründet werden, ob die bestehenden Ansätze bereits die Integration von Prozessen unterstützen oder ob sie geeignet sind, um diese erweitert zu werden. Um ein Modell zur Beschreibung mobiler Prozesse zu erhalten, wurden bestehende Prozessbeschreibungssprachen untersucht.

In diesem Zusammenhang zu betrachtende Middleware-Ansätze sind unter anderen: *Nexus* [4], als Plattform für kontextbewusste Anwendungen auf Basis eines föderierten und globalen räumlichen Kontextmodells; das *Projekt m3* [6], welches einen Broker zum Anpassen der Übertragung von Protokoll- und Nutzdaten bezüglich des aktuellen Kontexts bereitstellt; *Gaia* [8, 7], als Meta-Betriebssystem zur Unterstützung der Entwicklung und Ausführung von mobilen Anwendungen in „Ubiquitous Habitats“; *RCSM* [10], welches eine Middleware für situationsbewusste Software in Pervasive-Computing-Umgebungen darstellt und *PCOM* [3], als komponentenbasiertes, leichtgewichtiges und anpassungsfähiges System für verteilte Anwendungen mit explizit for-

mulierten Abhängigkeiten.

Um schließlich zu einem eigenständigen Prozessmodell zu kommen, müssen zudem unter anderem folgende Prozessbeschreibungssprachen betrachtet werden: *XPDL* [5, 9], als eine technologieunabhängige Definitionssprache eines Meta-Modells zum Austausch von Prozessen sowie die Sprache *BPEL4WS* [1], als Ergänzung des Web-Service-Protokollstapels, um Prozessbeschreibungen auf Basis des Orchestration-Paradigmas definieren und ausführen zu können. Im Gegensatz dazu basiert das *WSCI* [2] auf dem Choreography-Paradigma und ist als interne Erweiterung des WSDL-Standards konzipiert, die lediglich die Beteiligung einzelner Dienste aus deren lokaler Sicht an einem Prozess beschreibt.

3 Prozessintegration in mobile Umgebungen

Die Untersuchung von bestehenden Middleware-Systemen im Bereich des Mobile Computings und von verbreiteten Prozessmodellen (vgl. Abschnitt 2) hat zum dem im folgenden Abschnitt dargelegten Konzept zur Integration mobiler Prozesse geführt. Wie dieses Konzept im DEMAC-Projekt in eine Systemarchitektur umgesetzt wurde, wird dann im Abschnitt 3.2 beschrieben.

3.1 Konzept zur Integration mobiler Prozesse

Der Austausch und die verteilte Ausführung von Prozessen stellen an eine zugrunde liegende Systeminfrastruktur Anforderungen, die aus unserer Sicht von keinem der bestehenden Ansätze ausreichend unterstützt werden. Mobile a priori unbekannte Prozesse, deren Abarbeitung von nicht-funktionalen Attributen determiniert wird, benötigen ein möglichst allgemeines und generisches Kontextmodell. Dieses muss neben den nicht-funktionalen Parametern – wie zum Beispiel *Quality-of-Service-Parameter* oder Ortsangaben – auch eine Unterstützung zum Auffinden und Einbinden von sehr abstrakt beschriebenen Diensten enthalten. Vor allem dieser Aspekt der nicht ausreichenden Kontextmodelle macht es erforderlich, eine eigene, an die Bedürfnisse der Prozessintegration angepasste Middleware zu konzipieren und zu realisieren (vgl. Abschnitt 3.2).

Hierbei sollen mobile Systeme über einen möglichst einfachen asynchronen Kommunikationsmechanismus gekoppelt werden. Es wird dabei sowohl die anwendungs- als auch die ereignisgesteuerte Kommunikation unterstützt. Die dabei verwendete Adressierung des Transportmechanismus ist unabhängig bezüglich der darunter liegenden Transportprotokolle. Dies ermöglicht es, dass das tatsächlich verwendete Protokoll gegenüber den Anwendungen verborgen bzw. gezielt durch Adaptionsstategien festgelegt werden kann. Es ist damit außerdem möglich, beim Ausfall eines Übertragungskanals (für die

Anwendung transparent) auf eine andere Verbindung zum Kommunikationspartner auszuweichen. Weiterhin wird als weitere Basiskomponente ein allgemeines und generisches Kontextmodell benötigt, welches in der Lage ist, neben beliebigen Attributen und Attributhierarchien auch Dienste und Dienshierarchien zu beschreiben. Damit lassen sich Dienstklassen definieren, die semantisch sowie in der Aufrufsyntax äquivalent sind und sich lediglich in der technologischen Umsetzung unterscheiden. Diese im Kontextmodell inhärente Abstraktion von Dienstklassen ermöglicht es im Prozessmodell, Dienste sehr effizient zu beschreiben. Ein Prozessinterpreter umfasst als drittes Basiselement das Prozessmodell und den Prozesslebenszyklus der mobilen Prozesse. Dieser ist für die Ausführung des mobilen Prozesses und die Einhaltung der durch nicht-funktionale Aspekte vorgegebenen Rahmenbedingungen verantwortlich. Dazu muss er während der Laufzeit des Prozesses die Ausführung kontinuierlich an die jeweiligen Kontextbedingungen anpassen und ggf. an andere Geräte delegieren.

Da die bestehenden Prozessmodelle nicht für den Einsatz in einer mobilen Umgebung konzipiert wurden, sind sie meistens zwar sehr ausdrucksmächtig, bieten aber kaum die Möglichkeit, ihre Komplexität durch Abstraktionen zu reduzieren. Zudem sind viele Ansätze nicht technologieunabhängig und stellen keine Unterstützung zur Formulierung nicht-funktionaler Aspekte bereit. Deshalb soll hier ein geeignetes Prozessmodel konzipiert werden, welches besser für den Einsatz in einem Middleware-System für mobile Umgebungen geeignet ist. Dieses ist ähnlich wie das XPDL-Modell als technologieunabhängiges Meta-Prozessmodell angelegt. Es umfasst vor allem ein sehr allgemeines und effizientes Verfahren zur Beschreibung der im Prozess enthaltenen Aktivitäten auf Basis abstrakter Dienstklassen. Zudem wurde die Möglichkeit geschaffen, nicht-funktionale Parameter zu definieren, die als Rahmenbedingung während der Ausführung gelten. Dies führt dazu, dass auch delegierte Prozesse noch der Intention des Benutzers bzw. der Benutzerin folgen.

3.2 Architektur des DEMAC-Ansatzes

Die auf Grund der Analyse bestehender Ansätze und Prozessmodelle entwickelte DEMAC-Systemarchitektur integriert mobile Prozesse nahtlos in eine adaptive Infrastruktur für das Mobile Computing: Sie besteht im Wesentlichen zzt. aus vier Komponenten – jeweils einer für die Kommunikation mit Push- und mit Pull-Semantik, einer für das Kontext-Management und einer für die Prozessausführung:

Der *Asynchronous Transport Service* stellt die Kommunikationsbasis für das DEMAC-System dar und kapselt als nachrichtenorientierter Transportmechanismus die unterliegenden konkreten Transportprotokolle, wie TCP/IP, Bluetooth oder IrDA. Ein eigenes Adressierungsschema hilft dabei unabhängig

von den Adressen der Transportprotokolle zu sein.

Darauf aufbauend realisiert der *Event Service* den proaktiven Versand von Ereignisnachrichten zum Bekannt machen von Attributänderungen innerhalb und außerhalb der lokalen Infrastruktur.

Der *Context Service* hat die Aufgabe, jegliche Information über den Kontext des Gerätes zu sammeln und zu verwalten. Dabei wird das lokale Wissen mit Hilfe des Föderationsprinzips um Wissen über Geräte in der Umgebung erweitert. Um Dienste und Geräte in der Umgebung finden zu können, ist ein verteilter Verzeichnisdienst integriert, der seine Daten mittels Peer-to-Peer-Techniken erwirbt.

Die Integration mobiler Prozesse in die DEMAC Architektur wird durch den *Process Service* realisiert, der zum einen aus der Prozessbeschreibungssprache DPDL (*DEMAC Process Description Language*) und zum anderen aus einer Ausführungsumgebung für mobile Prozesse besteht. Durch die entwickelte Beschreibungssprache werden neben dem Prozess auch nicht-funktionale Parameter kompakt beschrieben. Diese stellen Randbedingungen für die verteilte Ausführung dar und berücksichtigen somit die Intentionen des Nutzers auch außerhalb seines Verantwortungsbereichs.

Die Ausführungsumgebung ist modular aufgebaut und gruppiert sich um den zentralen Prozessinterpreter. Dieser stützt sich auf einen *erweiterten Prozesslebenszyklus*, in dem zusätzliche Zustände eingefügt sind, um einen laufenden mobilen Prozess sicher an andere Geräte übertragen zu können. Dabei werden während der Ausführung und Übertragung stets die nicht-funktionalen Aspekte des Nutzers berücksichtigt. Die in der Beschreibungssprache abstakt gehaltenen Aktivitäten werden erst möglichst spät und in einem zweistufigen Verfahren zu konkreten Diensten aufgelöst und dann ausgeführt (*späte Bindung*). Damit ist es im Prinzip möglich, Umsetzungen der abstrakten Dienste in vielen Technologien bereit zu stellen, deren Nutzung dann lediglich durch die beherrschten Verfahren des einzelnen Geräts eingeschränkt wird.

4 Zusammenfassung und Ausblick

In diesem Papier wird argumentiert und erläutert, dass die explizite Unterstützung von Prozessen im Bereich des Mobile Computings dazu führen kann, dass die Fähigkeiten von verteilten Anwendungen in mobilen Umgebungen wesentlich erweitert werden. Dabei werden durch das Zusammenwirken der Konzepte des Kontexts, der Kollaboration und des Delegierens die Prozesse selbst (bzw. ihre Ausführung) mobil. Aufbauend auf dem Konzept für derartige Prozesse (bzw. Prozessausführungen) wird eine entsprechend erweiterte Middleware-Unterstützung vorgeschlagen und gezeigt, wie diese im Rahmen des Projektes DEMAC umgesetzt wird.

Eine prototypische Implementierung des vorgestellten asynchronen Trans-

port-Dienstes und des Event-Dienstes sind innerhalb des DEMAC-Frameworks zzt. bereits realisiert und in einer heterogenen und stark verteilten Systeminfrastruktur auf ihre Eignung getestet worden. Zudem sind die Prozessbeschreibungssprache DPDL spezifiziert und ein Prozessinterpreter dafür implementiert worden. Die dabei gemachten Erfahrungen haben bisher die Erwartungen erfüllt; demnächst sollen u.a. diese Konzepte im Rahmen einer exemplarischen Realisierung und Bewertung typischer Beispielszenarios noch weiter evaluiert werden.

Literatur

[1] T. Andrews, F. Curbera, H. Dholakia, Y. Goland, J. Klein, F. Leymann, K. Liu, D. Roller, D. Smith, S. Thatte, I. Trickovic, and S. Weerawarana. Business process execution language for web services version 1.1. Specification, IBM, BEA Systems, Microsoft, SAP AG, Siebel Systems, 2003.

[2] A. Arkin, S. Askary, S. Fordin, S. Jekeli, S. Kawaguchi, D. Orchard, S. Pogliani, K. Riemer, S. Struble, P. Takacsi-Nagy, I. Trickovic, and S. Zimek. Web service choreography interface (wsci) 1.0. Specification NOTE-wsci-20020808, World Wide Web Consortium, 2002.

[3] C. Becker, M. Handte, G. Schiele, and K. Rothermel. Pcom - a component system for pervasive computing. In *PERCOM '04: Proceedings of the Second IEEE International Conference on Pervasive Computing and Communications (PerCom'04)*, pages 67–76, 2004.

[4] F. Dürr, N. Hönle, D. Nicklas, C. Becker, and K. Rothermel. Nexus–a platform for context-aware applications. In J. Roth, editor, *1. Fachgespräch Ortsbezogene Anwendungen und Dienste der GI-Fachgruppe KuVS*, 2004.

[5] R. Norin and M. Marin. Workflow process definition interface – xml process definition language. Specification WFMC-TC-1025, Workflow Management Coalition, 2002.

[6] A. Rakotonirainy, J. Indulska, S. W. Loke, and A. Zaslavsky. Middleware for reactive components: An integrated use of context, roles and event based coordination. In *Proceedings of the IFIP/ACM International Conference on Distributed Systems Platforms*, number 2218 in LNCS, pages 77–98. Springer Verlag, 2001.

[7] M. Román, C. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, and K. Nahrstedt. Gaia: A middleware platform for active spaces. *SIGMOBILE Mob. Comput. Commun. Rev.*, 6(4):65–67, 2002.

[8] M. Román, C. K. Hess, R. Cerqueira, A. Ranganathan, R. H. Campbell, and K. Nahrstedt. Gaia: A middleware infrastructure to enable active spaces. *IEEE Pervasive Computing*, pages 74–83, 2002.

[9] W. van der Aalst. Patterns and xpdl: A critical evaluation of the xml process definition language. Technical Report FIT-TR-2003-06, Queensland University of Technology, 2003.

[10] S. S. Yau, D. Huang, H. Gong, and S. Seth. Development and runtime support for situation-aware application software in ubiquitous computing environments. In *Proc. 28th Annual Int'l Computer Software and Application Conference (COMPSAC 2004)*, pages 452–457, 2004.