

# mobileISM: Ein Framework zur adaptiven Unterstützung der Nutzung mobiler Dienste

Elisabeth Heinemann, Erich Ortner, Nicolas Repp

Technische Universität Darmstadt  
D-64289 Darmstadt, Deutschland

{heinemann, ortner, repp}@winf.tu-darmstadt.de, URL: <http://www.winf.tu-darmstadt.de>

## Abstract

Der folgende Artikel deckt verschiedene wissenschaftliche Bereiche ab und ist daher als interdisziplinär einzuordnen. Im Einzelnen sind dies:

- eine Management-Umgebung für mobile Dienste,
- ein Framework bzw. eine Plattform (E-NOgS) als Referenzarchitektur und theoretische Basis zur Entwicklung interaktiver Anwendungen und
- mobileISM (mobile intelligent service manager) als dazugehörige implementierte Lösung.

Er ist als ein Transfer-Beitrag – aus der Wissenschaft für die Praxis – verfasst.

## 1 Die Herausforderung

Das Internet, so wie wir es kennen, scheint – auch nach Meinung vieler Experten – langsam an Bedeutung zu verlieren und schrittweise von einer mehr serviceorientierten Netzwerkvariante wie z.B. der Web Service Technologie ersetzt zu werden. Denkt man also an ein Internet, das nicht nur aus statischen Web-Seiten besteht, so steckt in dieser Idee sehr viel Potential für die Entwicklung mobiler interaktiver Anwendungen, denn mobile Endgeräte könnten verschiedene Dienste kombinieren, um sich rasch verändernden Anforderungen quasi „on-the-fly“ anzupassen. Führt man diesen Gedanken konsequent weiter, so sind ein mögliches Ergebnis entsprechender Entwicklungen kleine (low-tech), aber dennoch mächtige, mobile Endgeräte, die ihre Stärke aus der Fähigkeit ziehen, verteilte Dienste über Computer Netzwerke nutzen zu können. Wahre Mobilität basiert demzufolge offensichtlich auf der Netzwerkfähigkeit mobiler Endgeräte und der entsprechenden Software (z.B. Service Handling, siehe Kap. 2.1).

Zu Anfang dieses neuen Jahrhunderts der Entwicklung von Informationstechnologien darf ein System-Architekt aber niemals den immer noch wichtigsten Faktor eines IT Systems aus den Augen verlieren: den Menschen, der es (be)nutzt. Stellen wir uns einen in zweierlei Hinsicht mobilen User vor. Er ist beweglich sowohl hinsichtlich seines Standorts als auch bezüglich der Inanspruchnahme angebotener Dienste. Aufgrund dieser Art von Mobilität erhält er zahlreiche Service- und Informationsangebote (so genannte *location based services*), von denen aber bei weitem nicht alle auch tatsächlich relevant oder interessant für ihn sind.

Wie sollte die benutzte Anwendung auf diesen Um-

stand reagieren? Die naheliegendste Möglichkeit besteht sicherlich darin, den Nutzer selbst entscheiden zu lassen, welche Informationen und/oder Dienste er tatsächlich nutzen möchte. Aber das kann schnell zu einem *Information-Overflow*, also einem Überangebot von Informationen führen. Abhängig von der Menge der Informationen wird der Anwender letztendlich gar nichts auswählen, da er von der ihm zur Verfügung stehenden Angebotsmenge „erschlagen“ wird. Eine zweite Möglichkeit besteht darin, das mobile Endgerät *selbsttätig* [Pe01] entscheiden zu lassen, welche Informationen und Dienste dem Benutzer tatsächlich offeriert werden (*smart filtering*, siehe auch Kap 2.2).

In jedem Fall sollte es das wichtigste Anliegen eines Systemarchitekten sein, eine mobile Anwendung dergestalt zu entwickeln, dass sie fähig ist, einen Endanwender abhängig von der Situation, in der er sich gerade befindet, automatisch und bestmöglich zu unterstützen.

## 2 Der Kontext

Ziel unserer Forschungsarbeit ist es, eine Umgebung zu schaffen, die Anwendungsentwickler in die Lage versetzt, benutzerorientierte und –unterstützende interaktive Anwendungen für mobile Endgeräte zu entwickeln. Dies schließt verschiedene Aspekte mobiler (interaktiver) Anwendungen ein, u.a. natürlich auch auftretende Probleme.

Doch zunächst müssen wir die Schlüsselemente unserer Umgebung definieren. Sofern wir mobile Systeme als mobile Dienste betrachten, die über die Fähigkeit verfügen, diverse andere Dienste via Netzwerke zu einer beliebigen Zeit an einem beliebigen Ort zu nutzen, sollten wir uns mit diesen auch etwas genauer beschäftigen. Unsere *Management-Umgebung* [KR03] kennt verschiedene Arten von Diensten, deren Unterschied darin besteht, inwieweit sie vor Ort genutzt werden und ob sie Ortsparameter als Input für ihre Operationen verwenden können (siehe Abb. 1).

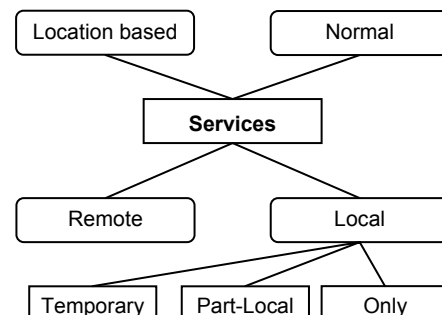


Abbildung 1: Mögliche Dienste (Services)

Alle in Abb. 1 gezeigten Dienstetypen sind in Verbindung mit einer mobilen Umgebung gesehen sinnvoll. Die Unterscheidung von Diensten, die mit bzw. ohne standortbezogene Funktionalität arbeiten, ist in einer mobilen Umgebung trivial. Wichtiger hingegen ist die Unterscheidung von *Local* und *Remote Services*. Unser Ansatz sieht beide Varianten vor. Dienste (in Form einer Anwendung im herkömmlichen Sinne) können auf einem mobilen Endgerät aus der Ferne (*remote*) installiert werden. Für den Nutzer macht es keinerlei Unterschied, ob ein Dienst lokal oder auf einem fernen Server zur Verfügung steht. Local Services können überdies dauerhaft oder temporär installiert werden, wahlweise mit ihrer kompletten Programmlogik direkt auf dem Endgerät oder einer teilweisen Auslagerung auf ein Remote System.

Die hier beschriebene Management-Umgebung (siehe Abb. 2) ist in der Lage eine große Menge solcher Dienste, remote ebenso wie lokal, selbsttätig zu verwalten. Sie erlaubt einem mobilen Nutzer überdies das Arbeiten mit persönlichen Präferenzen, also eine benutzerorientierte Verwendung solcher Dienste in Abhängigkeit von Vorlieben und Abneigungen. Darüber hinaus ermöglicht das mobile Framework (siehe Abb. 2) nicht nur die Nutzung von Diensten, sondern auch deren Abrechnung und Bezahlung auf eine zwar sichere, aber dennoch flexible Art und Weise. Unsere Umgebung beachtet des weiteren verschiedene Interessenvertreter innerhalb eines mobilen Anwendungsfeldes (Endanwender, Entwickler und Anbieter von Inhalten) sowie deren individuelle Anforderungen. In den folgenden Abschnitten werden wir einige dieser Aspekte genauer betrachten.

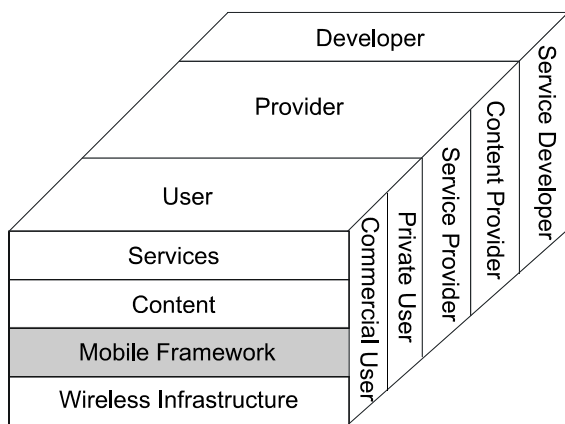


Abbildung 2: Verbindung der Themenfelder zu einer Management-Umgebung

## 2.1 Service Handling

Bestimmte Typen von Diensten benötigen verschiedene Mechanismen, um mit ihnen arbeiten zu können. In einem mobilen Szenario ist deshalb Ortstransparenz (*location transparency*) ein wichtiger Aspekt. Das hier vorgestellte Framework gestattet die Ausführung verschiedener Protokolle zum Auffinden von Remote Services (z.B. Microsoft Universal Plug and Play oder Salutation). Es ist weiterhin möglich, sich mit einigen proprietären Servern in Verbindung zu setzen, um das komplette Spektrum möglicher Dienste auszunutzen wie beispielsweise das derjenigen, die vor der Nutzung installiert werden müssen. Dienste werden in einem Verzeichnis ähnlich UDDI (Universal Description,

Discovery and Integration) verwaltet, d.h. es existiert für jeden beliebigen Service in einem Repository eine entsprechende Beschreibung. Darüber hinaus werden installierbare Dienste zur weiteren Benutzung auf einem Speichermedium gesichert.

## 2.2 Smart Filtering

Um der bereits erwähnten Informationsflut Herr zu werden, unterstützt das mobile Framework (siehe Abb. 2) verschiedene Arten eines automatischen Entscheidungs- und Filtervorgangs, das sogenannte *Smart Filtering*. Im Augenblick werden hierfür Bayesche Netze und Vorhersagen eingesetzt. Das Framework kann dadurch beispielsweise selbsttätig (proaktiv) entscheiden – basierend auf der Nutzungshäufigkeit eines bestimmten Dienstes an einem bestimmten Ort zu einer bestimmten Zeit – ob ein spezifischer Dienst aus einer Menge vieler es wert ist, dem User angeboten zu werden.

Darüber hinaus kann das Framework jegliche Information – nicht nur Service Informationen – die der Nutzer erhält, klassifizieren. Auch dies basiert wiederum auf Bayeschen Netzen. Das Framework analysiert in diesem Fall die Aktivitäten des Users und die Informationen, auf die er zurückgreift. Beispielsweise möchte er einen Service nutzen, um seine Termine zu verwalten. Das Framework filtert die auf seinen persönlichen Präferenzen basierenden Suchergebnisse und komprimiert bzw. reduziert die so gewonnenen Informationen, damit der Nutzer sich schneller und einfacher entscheiden kann. Zukünftige Versionen dieses Frameworks, das einmal die Funktionalität von NOGS (New Organon {Server, Service, Servant}) besitzen soll [OO02], werden Entscheidungsalgorithmen effektiver und genauer unterstützen.

## 2.3 Sicherheit

Das Framework konzentriert sich auf drei wesentliche Sicherheitsaspekte:

1. es erzwingt die Verschlüsselung jeglicher Kommunikation,
2. erlaubt nur die Ausführung signierter Dienste und
3. stellt sicher, dass keine unnötigen Daten den Client „verlassen“.

Dieser Ansatz garantiert ein höchstes Maß an Sicherheit und Anonymität für den Benutzer, denn alle Aktionen, die persönliche Daten des Users betreffen (Profilerstellung und Filtern von Diensten und/oder Informationen) werden direkt auf dem Client ausgeführt.

Während des Entwurfs des mobilen Frameworks wurde besonderer Wert auf eine konsequent universelle Konstruktion gelegt. Auf diese Art und Weise ist es sehr einfach Verschlüsselungen – falls erforderlich – zu ändern (z.B. könnten einige Verschlüsselungsalgorithmen unsicher werden). Alle Sicherheitsaspekte müssen konform mit den entsprechenden Rechtsvorschriften beteiligter Länder sein.

## 2.4 Bezahlung

Um ein solches Framework erfolgreich zu etablieren, ist es wichtig und notwendig vernünftige Konzepte für Abrechnung und Bezahlvorgang zu integrieren. Hier gilt es insbesondere die „kleineren“ Geschäfte im Bereich von unter zehn Euro abzudecken. Folgende Aspekte sollten besondere Berücksichtigung finden:

- Traditionelle Konzepte wie z.B. Kreditkarten versagen hier auf grund ihrer hohen Transaktionskosten.
- Verzögerungszeiten bei Zahlungsbestätigungen müssen bestmöglich reduziert werden.
- Bezahlvorgänge müssen so einfach wie möglich gehalten werden (z.B. Verhinderung der Mehrfachabfrage von PIN oder Passwort).

Um die aufgeführten Herausforderungen zu bewältigen, erscheint es vernünftig, so genannte kleine (z.B. Net-Bill, Mondex, Cybercoin) und Mikro-Zahlungsvarianten (Minipay, Millicent, MicroMint) zu integrieren. Der Vorteil dieser Techniken liegt in ihren geringen Gemein- und Transaktionskosten, einer einfachen Handhabung und einem hohen Grad an Sicherheit. Ein weiteres Plus liegt in der Tatsache begründet, dass Teile des Systems bereits existieren und kein spezifisches Re-Design notwendig wird.

### 3 E-NOgS: Eine Referenzarchitektur für stationäre und mobile generische Dienste

Um nun die bereits genannten anwendungsübergreifenden Anforderungen tatsächlich zu berücksichtigen wurde ein *Middleware-Framework* [OO02] einbezogen, das entsprechende Funktionalitäten bereitstellt. Die generische Referenzarchitektur *E-NOgS* (Electronic New Organon {Server, Service, Servant}) ist als ein grundlegendes Werkzeug (griech. *Organon*) zur Entwicklung mobiler Anwendungen konzipiert. Sie eignet sich gleichermaßen für den Bau großer (z.B. elektronischer Marktplatz) als auch kleiner Anwendungen (z.B. mobile Agenten). Im Hinblick auf ihre generischen Funktionen wurde sie zunächst sprachlogisch, d.h. basierend auf den jeweils vorherrschenden Sprachhandlungstypen aus den Gebrauchssprachen der Anwender rekonstruiert [Au62].

Tabelle 1 beschreibt unseren Ansatz näher und bringt besonders die sprachkritische Fundierung der Architektur von E-NOgS zum Ausdruck.

Auf der Grundlage des Konzepts von „Schema und Ausprägung“ bzw. „Sprachhandlung“ und dem der Bildung von „Sprachebenen“ und „Sprachräumen“ wurden aus dem konkreten Spracheinsatz in Wirtschaftsgemeinschaften bzw. Unternehmen generische und spezifische Sprachhandlungen rekonstruiert. Generische Sprachhandlungen, die noch keinen spezifischen Anwendungsbereich haben, führen in der Informatik und Wirtschaftsinformatik zu den so genannten *Basissystemen*. In dem zu Grunde liegenden komponentenorientierten Aufbau wird jede generische Funktionalität als ein solches Basissystem (z.B. Datenbank-Management-System, Workflow-Management-System) realisiert. Konkrete Funktionen können dadurch flexibel eingebunden werden und mögliche Entwicklungen müssen nicht zwangsweise das komplette Framework als Grundlage heranziehen. Dieser Umstand macht E-NOgS insbesondere als Referenzarchitektur für den generischen Teil mobiler Anwendungen sinnvoll.

In einem Unternehmen können die beschriebenen Basissysteme erst benutzt werden, wenn die Anwendung – also die spezifische Sprachhandlung – dazu entwickelt wurde. Beispielsweise können zu dem generischen rechnerunterstützten Sprachhandeln „Datenverwaltung“ die spezifischen rechnerunterstützten Sprachhandlungen „Kundendatenverwaltung“ als Datenbankapplikationen entwickelt werden. Tabelle 1 zeigt in der letzten Zeile für die verschiedenen generischen Funktionen (Basissysteme) die Einsatzgebiete auf und unterbreitet für die Entwickler Vorschläge, welche Logik jeweils als Basis zur Entwicklung der Lösungen heran zu ziehen ist (siehe Tab. 1, 4. Zeile).

Systemtyp	Datenbank-Management-Systeme	Workflow-Management-Systeme	Entscheidungsunterstützungssysteme	Kommunikationsmanagement-Systeme	...	Reflexionsunterstützungssysteme
<b>Merkmal</b>						
<b>Vorherrschender Sprachhandlungstyp</b>	äußern/ verstehen	veranlassen/ überwachen	fragen/ antworten	zustimmen/ ablehnen	...	einschätzen/ bereitstellen
<b>Ebenenpaar*)</b>	Schema/ Ausprägungen	Steuerung/ Ausführung	Prämissen/ Konklusion	Transformation/ Interaktion	...	Metaebene/ Objektebene
<b>primärer Modellierungsgegenstand</b>	Dinge	Geschehnisse	Folgerungen	Verständigungen	...	Erwägungen
<b>Logik als Basis für die Systementwicklung</b>	Standardlogik (Relationenkalkül, etc.)	Modallogik (Imperativlogik, etc.)	Syllogistik (Fragelogik, etc.)	Interaktionslogik (dialogische Logik, etc.)	...	Metatheorien (Metalogik, etc.)
· ·						
<b>Einsatzgebiete</b>	Produktions- und Administrationsaufgaben	Führungs- und Kontrollaufgaben	Analyse- und Entscheidungsaufgaben	Übersetzungs- und Vermittlungsaufgaben	...	Konstruktions- und Inventionsaufgaben

\*) Anmerkung: zu den Ebenen „Steuerung/Ausführung“, „Prämissen/Konklusion“, „Transformation/Interaktion“ und „Metaebene/Objektebene“ ist das Ebenenpaar „Schema/Ausprägungen“ orthogonal.

Tabelle 1: Gegenüberstellung der sprachlogisch rekonstruierten Basissysteme

## 4 Beleg für das Konzept: mobileISM

Basierend auf dem hier vorgestellten Framework haben wir eine Beispielanwendung entwickelt, die das beschriebene Konzept hinsichtlich seiner Tauglichkeit und Einsatzmöglichkeiten belegen soll. Diese Anwendung wurde innerhalb eines regulären Praktikums an der Technischen Universität Darmstadt von etwa zwanzig Studenten der Wirtschaftsinformatik im Hauptstudium entwickelt und stieß auf der CeBIT 2003 in Hannover auf reges Interesse.

*MobileISM* (mobile intelligent service manager, siehe Abb. 3) ist eine Microsoft PocketPC 2002 basierte Softwarelösung, die momentan den Fokus auf die Aspekte Service Handling (Workflow-Management-System, siehe Tab. 1) und Smart Filtering (Reflexionsunterstützungssystem, siehe Tab. 1) legt. Sie integriert ortsabhängige Funktionalität (interaktive Anwendungen) mit Service Handling und dem Potential von Web Service Technologien sowie einem Smart Filtering auf der Basis eines Repository-Systems, in dem die Services (Anwendungen) beschrieben sind. Letzteres entscheidet entweder proaktiv welche Dienste das tägliche Leben seines Nutzers erleichtern oder unterbreitet diesbezüglich konkrete Vorschläge.

Aus Tabelle 1 (E-NOgS Funktionalität) versucht die *Inference-Engine* (siehe Abb. 3) vor allem die generischen Funktionen Entscheidungs- und Reflexionsunterstützung zu erfüllen. Durch Reflexion wird ein Schlussfolgern über mehrere Sprachebenen ermöglicht. Dadurch können von einer Metasprachebene wie z.B. einem Repository aus, Schemata (Dienste) für das (Sprach-)Handeln in einem Anwendungsbereich (Objektsprachebene) „automatisch“ bereitgestellt werden. Nach dem Kauf eines Artikels beispielsweise bringt mobileISM den Service „Bezahlung“ zum Einsatz und überwacht dessen Vorgangsausführung (Workflow-Management-System).

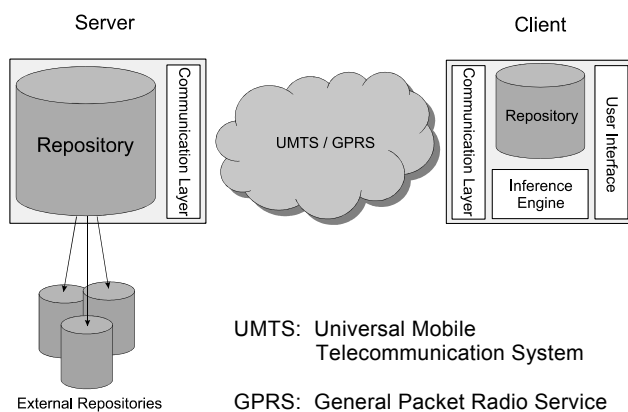


Abbildung 3: mobileISM

Auf mobileISM können verschiedene andere Anwendungen aufgesetzt werden wie beispielsweise Navigationsdienste und andere location based services.

## 5 Zusammenfassung

In dem vorliegenden Artikel haben wir verschiedene Aspekte hervorgehoben und zu einer Management-Umgebung zusammengefasst, von denen wir überzeugt sind, dass sie für das Mobile Computing Relevanz ha-

ben. Wir führten ein Informations- und Dienst-Framework (mobileISM) zum Management mobiler Services ein, das den Benutzer in vielen verschiedenen Situationen selbsttätig unterstützen soll. Die Hauptaspekte waren hierbei Smart Filtering, Service Handling, Sicherheit und Bezahlung. Eine sehr wichtige, aber auch kritische Rolle spielt aus unserer Sicht das Vertrauen des Nutzers.

Auf diesem Gebiet stehen wir jedoch noch ganz am Anfang. Bis wir wirklich einmal von der „ubiquitous beauty of user-aware software“ [Pa01] schwärmen können, müssen noch viele sprachlogische (z.B. non-verbale Kommunikation) und technische Aspekte (z.B. multimediale Schnittstellen) geklärt werden.

## 6 Literatur

- [Au62] Austin, J.L.: How to Do Things with Words. Oxford University Press, Oxford, Cambridge (Mass.), 1962
- [KR03] Karg, L; Repp, N.: Mobile Computing – A Service Framework. Fachgebiet Wirtschaftsinformatik I, Technische Universität Darmstadt, unveröffentlichter Beitrag, Darmstadt 2003.
- [OO02] Ortner, E; Overhage, S.: E-NOgS: Ein komponentenorientiertes Middleware-Framework für E-Commerce-Anwendungen. In: Weinhardt, Ch.; Holtmann, C. (Hrsg.): E-Commerce – Netze, Märkte, Technologien. Physiker Verlag, Heidelberg 2002. S. 241-251.
- [Pa01] Pancake, C.: The Ubiquitous Beauty of User-aware Software; Communications of the ACM archive, Volume 44, Issue 3; 2001