Selbstorganisierende Infrastrukturen für Ambient Services

Klaus Herrmann

Universität Stuttgart
Institut für Parallele und Verteilte Systeme (IPVS)
Universitätstr. 38, 70569 Stuttgart
klaus.herrmann@acm.org

Abstract: Die Vision von *intelligenten Umgebungen* (Ambient Intelligence – AmI) als neues Paradigma für die Unterstützung mobiler Benutzer bei alltäglichen Tätigkeiten rückt derzeit zunehmend in den Fokus Europäischer Forschungsaktivitäten. Ein hoher Grad an Autonomie auf Seiten der unterstützenden Software-Systeme ist hierfür eine unabdingbare Voraussetzung. Derartige Systeme müssen in der Lage sein, sich selbst ohne steuernde Eingriffe von Benutzern an deren Aktivitäten und Verhalten anzupassen. Entsprechende Systeme sind derzeit jedoch weitgehend unerforscht. Diese Arbeit stellt den ersten umfassenden Ansatz für eine selbstorganisierende Plattform zur Unterstützung von so genannten *Ambient Services* dar – Diensten, die es dem mobilen Benutzer gestatten, mit seiner unmittelbaren Umgebung zu interagieren. Diese als *Ad hoc Service Grid* (ASG) bezeichnete Plattform kann spontan in beliebigen Umgebungen (z.B. Einkaufszentren, Baustellen, Messen, etc.) eingesetzt werden, um dort entsprechende Dienste anzubieten. In der Arbeit schlagen wir zunächst ein Modell für eine entsprechende Infrastruktur aus spontan vernetzten Rechnerknoten vor. Diese ist modular aufgebaut und im Einsatz flexibel erweiterbar. Der eigentliche Fokus liegt jedoch auf der Entwicklung und Evaluierung einer Reihe von Basis-Algorithmen und Protokollen für das selbstorganisierte Betreiben dieser Infrastruktur. Hier konzentrieren wir uns auf die Platzierung von Ambient Service-Instanzen, deren Auffinden (Discovery und Lookup), und auf die Erhaltung der Konsistenz einer Gruppe verteilt laufender Replikate eines solchen Dienstes. Diese Mechanismen legen zusammen eine essenzielle Grundlage für die Weiterentwicklung intelligenter Umgebungen.

1 Einleitung

In den letzten Jahren hat die Vision *intelligenter Umgebungen* (Ambient Intelligence – AmI) [DBS+01] bemerkenswerte Forschungsanstrengungen hervorgebracht. Die grundlegende Idee von AmI ist der flächendeckende drahtlose Zugang mobiler Benutzer zu Diensten, welche sie automatisch in ihren täglichen Aktivitäten unterstützen, ohne ihnen die Administration der entsprechenden Infrastrukturen aufzubürden.

Ein essenzieller Baustein derartiger AmI-Infrastrukturen sind Systeme, die Benutzern Zugang zu lokalen Diensten bieten. Solche Dienste ermöglichen die Interaktion zwischen dem Benutzer und seiner Umgebung. So ist z.B. denkbar, dass in einem Einkaufszentrum

Dienste für Navigation, Preisvergleiche und Routenplanung anhand eines elektronischen Einkaufszettels u.v.m. durch Ambient Services erbracht werden. Im Sinne der übergeordneten AmI-Vision können diese lokalen Infrastrukturen auf globaler Ebene verbunden und mit global verfügbaren, Benutzer-relevanten Informationen angereichert werden.

In diesem Beitrag beschreiben wir die Ad hoc Service Grid-Infrastruktur [Her06], welche umgebungsspezifische Ambient Services ermöglicht. Ein ASG kann mit wenig Aufwand in einer mittelgroßen Lokation eingerichtet werden. Es ist flexibel skalierbar und benötigt minimalen administrativen Aufwand. Hierbei machen wir uns Technologien der Spontanvernetzung zunutze, um diese Flexibilität auf Netzwerkebene zu erreichen. Um den administrativen Aufwand zu minimieren, führen wir eine Software-Schicht ein, die wir als *Serviceware* bezeichnen, und die Ambient Services innerhalb eines ASG selbstorganisiert betreiben kann. Alle Aspekte der Ausführung von Diensten und deren Anpassung an ein veränderliches Benutzerverhalten werden autonom von der Serviceware organisiert. Wir werden die Kern-Funktionen, -Algorithmen und -Protokolle beschreiben und zeigen, wie diese im Verbund den selbstorganisierten Betrieb eines ASG ermöglichen.

Der Rest des Papiers ist wie folgt gegliedert: In Abschnitt 2 diskutieren wir zunächst verwandte Arbeiten. Das ASG-Infrastruktur-Modell wird in Abschnitt 3 eingeführt, bevor wir in Abschnitt 4 einen Überblick über die Algorithmen und Protokolle geben. In Abschnitt 5 präsentieren wir unsere Schlussfolgerungen und geben einen Ausblick.

2 Verwandte Arbeiten

Es gibt derzeit zwei Hauptforschungsrichtungen im AmI-Bereich. Die erste befasst sich mit adäquaten Middleware-Lösungen zur Handhabung der Dynamik. So stellen z.B. LAICA [CFLZ05], SALSA [RFPV04], „agile agents“ [OOC+04] und der von Vallée et al. vorgeschlagene Web Service-basierte Ansatz Lösungen bereit, welche auf Agententechnologie beruhen. Hierbei wird meist angenommen, dass Agenten eine natürliche Fähigkeit besitzen, mit dynamischen Umgebungen adäquat umzugehen. Substanziiert wird dies jedoch nur selten.

Die zweite Forschungsrichtung im Bereich AmI stammt aus dem Bereich der *Service-Oriented Architectures* (SOA) und befasst sich mit den Problemen des Auffindens und der Komposition vom Ambient Services. Omnisphere [ROPD03] ist eine Architektur, die das Auffinden von Dienst-Komponenten und deren Komposition zu höherwertigen Diensten auf Basis sog. *Typed data flows* unterstützt. Hierbei werden Benutzer-Präferenzen, Geräte-Eigenschaften und Kontext-Informationen berücksichtigt. SodaPop [Hel05] ist ein System zur Komposition von Diensten aus Komponenten, welche sich auf den verschiedenen Geräten in einer Umgebung befinden. WSAMI Web Service Ambient Intelligence [IST+05] verwendet eine deklarative Sprache, um AmI-Systeme zu spezifizieren.

Keine der existierenden Arbeiten bietet jedoch einen umfassenden Ansatz, der vergleichbar mit der vorliegenden Arbeit ist. In unseren Untersuchungen betrachten wir vom Infrastrukturmodell über die Middleware bis hin zu integrierten Algorithmen und Protokollen alle grundlegenden Aspekte der Selbstorganisation in Ambient Intelligence-Systemen.

3 Das Ad hoc Service Grid-Modell

Es gibt auf Kommunikationsebene im Wesentlichen zwei Alternativen, um mobilen Benutzern Ambient Services anzubieten, welche die Interaktion mit der lokalen Umgebung unterstützen. Zum einen ist dies die klassische Mobilfunktechnologie. Allerdings ist hierbei die verfügbare Bandbreite stark beschränkt, und jegliche Kommunikation, auch wenn sie nur zwischen dem Benutzer und seiner unmittelbaren Umgebung stattfindet, ist kostenintensiv. Als zweite Alternative bietet sich die Abdeckung einer Lokation mit Access Points nach dem 802.11-Standard (WLAN) an. Solche Access Points dienen als drahtlose Verlängerung einer Draht-gebundenen Infrastruktur, während die Dienste zentral auf einem rechenstarken Server erbracht werden. Dieser Ansatz bietet hohe Bandbreiten, und die Kommunikation ist kostenlos. Allerdings ist er wegen seiner fest-verdrahteten Komponenten zum einem relativ unflexibel, und zum anderen ist die Verdrahtung mit unverhältnismäßig hohen Investitions-Kosten verbunden [KS03].

Keine der beiden genannten Alternativen ist als Kommunikations-Infrastruktur für Ambient Services geeignet. Deshalb schlagen wir als dritte Alternative das Ad hoc Service Grid vor, welches im Folgenden näher erläutert wird.

3.1 Das Modell

Das Konzept des ASG basiert auf Rechner-Knoten, die wir *Service Cubes* (im Folgenden auch *Cubes* oder *Knoten*) nennen. Ein Service Cube ist ein PC-artiger Rechner, der die Fähigkeit besitzt, sich spontan mit anderen Cubes drahtlos zu vernetzen. Des Weiteren bietet er eine gewisse Rechenkapazität. Er hat keine Peripherie-Geräte (Maus, Tastatur, Monitor, etc.) und wird permanent mit Strom versorgt. Um nun eine Lokation mit Service Cubes abzudecken, werden einige dieser Cubes in der Umgebung verteilt, so dass sie gemeinsam ein Ad hoc-Netzwerk aufbauen können. Da Service Cubes eigenständige Einheiten sind, ist ein ASG sehr modular: Module (Cubes) können jeder Zeit zum ASG hinzugefügt, aus ihm entfernt oder umplatziert werden, um die Struktur, die Gesamtrechenleistung und die verfügbare Netzwerkbandbreite spontan an veränderte Anforderungen anzupassen. Ausgedehnte Planungsphasen und hohe initiale Investitionen sind nicht notwendig. Darüber hinaus ist die Kommunikation innerhalb eines ASG kostenlos. Durch den modularen Aufbau und die Möglichkeit der Erweiterung im laufenden Betrieb ergeben sich des Weiteren neue Geschäftsmodelle für die Bereitstellung von ASGs. Von der zentralen Anschaffung durch einen Betreiber bis hin zur gemeinschaftlichen Anschaffung durch mehrere beteiligte Parteien (z.B. Ladenbesitzer im Einkaufszentrum) sind verschiedene Modelle denkbar. Auch das Vermieten von Cubes für den kurzfristigen Einsatz (z.B. auf Messen) wird durch dieses Konzept möglich. Benutzer, welche ein laufendes ASG verwenden möchten, werden einfach mit ihren mobilen Endgeräten durch das Mittel der Spontanvernetzung zum Teil des ASG-Netzwerkes und können so stets auf die laufenden Dienste zugreifen.

3.2 Die Drop-and-Deploy-Vision

Das ultimative Ziel der hier vorgestellten Technologie lässt sich am besten mit dem Begriff *Drop-and-Deploy* beschreiben: Jemand, der ein ASG zum Einsatz bringen möchte, muss lediglich eine angemessene Zahl von Service Cubes in der entsprechenden Lokation verteilen (Drop), so dass jeder Cube mindestens einen weiteren Cube in Funkreichweite hat. Nach dem Einschalten wird ein Netzwerk aufgebaut, und der Betreiber kann die benötigten Dienste auf einem beliebigen Cube *installieren* (Deploy). Im Weiteren übernimmt die Serviceware die Verteilung der Dienste, die Bindung zwischen Klienten und geeigneten Dienst-Instanzen und die Anpassung an veränderliche Bedingungen (z.B. Benutzerverhalten). Auf dem Weg hin zu diesem Ziel sind noch einige Hürden in unterschiedlichen Feldern der Informatik zu beseitigen. In der vorliegenden Arbeit konzentrieren wir uns darauf, hierfür die Grundlagen in Form von Algorithmen und Protokollen für den selbstorganisierten Betrieb zu legen.

4 Algorithmen und Protokolle zur Selbstorganisation eines ASG

Als Basis für die Selbstorganisation eines ASG haben wir drei grundlegende Funktionen identifiziert. Hierbei steht vor allem der Umgang mit der zu erwartenden Dynamik innerhalb eines ASG im Vordergrund. Benutzermobilität, wechselnde Dienstangebote und ein veränderliches Verhalten in der Benutzung von Diensten führen dazu, dass sich das System stets anpassen muss, um eine angemessene Leistung zu erbringen. Im Einzelnen werden folgende Aspekte untersucht:

1. **Selbstorganisierte Dienstverteilung:** Eine einzelne Instanz eines Dienstes, welche einem Service Cube fest zugeordnet ist, ist nicht ausreichend, um einer größeren Anzahl von Benutzern den Dienst in der gewünschten Qualität (z.B. bzgl. der Antwortzeit) zu erbringen. Temporäre Partitionierungen des Netzwerkes könnten zu Brüchen in der Verfügbarkeit führen. Anfragen müssten möglicherweise durch das gesamte Netzwerk geroutet werden. Letzteres ist vor allem in einem drahtlosen Multi-Hop-Netzwerk wie dem ASG ein Problem, da wertvolle Bandbreite verschwendet würde und die Antwortzeiten hoch wären. Daher sind die Replikation von Diensten und deren adäquate Platzierung innerhalb des ASG grundlegende Anforderungen. Die selbstorganisierte Dienstverteilung repliziert Dienste und positioniert die Replikate so, dass jeder Klient möglichst ein Replikat in unmittelbarer Nähe hat. Diese Verteilung wird darüber hinaus dynamisch an die aktuell herrschenden Anfragemuster angepasst.
2. **Service Discovery und Lookup:** Das Auffinden von Diensten wird durch die dynamische Replikation und Verteilung zu einer Herausforderung. Klienten müssen daher von einem entsprechenden Lookup Service, der mit der Dynamik im Netzwerk umgehen kann, verlässliche Informationen über adäquate Replikate erhalten können. Der Lookup Service selbst muss verteilt laufen und ein niedriges Aufkommen an zusätzlichen Netzwerknachrichten erzeugen, so dass durch das Lookup von

Diensten nicht das Netzwerk bereits ausgelastet wird.

3. **Datenkonsistenz zustandsbehafteter Dienste:** Dienste, die innerhalb eines ASG sinnvoll sind, tragen meist einen internen Zustand. Das heißt, sie speichern Daten, welche von Klienten verteilt gelesen und verändert werden können. Aus der Replikation von Diensten erwächst nun die Notwendigkeit entsprechender Protokolle für die Konsistenzerhaltung solcher replizierter Datenbestände. Diese Protokolle müssen ihrerseits der Dynamik innerhalb eines ASG Rechnung tragen.

In den folgenden Abschnitten werden wir die im Rahmen der Arbeit entwickelten Lösungen für jedes dieser drei Kernprobleme näher betrachten.

4.1 Selbstorganisierte Dienstverteilung

Wir haben einen adaptiven Algorithmus zur automatischen Replikation und Platzierung von Diensten entwickelt [HGM04], welcher vollständig verteilt und ohne Steuerung von außen von den einzelnen Replikaten ausgeführt wird. Dieser Algorithmus gestattet es Replikaten, sich selbständig zu replizieren (eine identische Kopie von sich zu erzeugen), zwischen Service Cubes zu migrieren und sich selbst aus dem System zu entfernen. Das Ziel dieses Algorithmus ist es, Replikate näher zu den anfragenden Klienten zu bewegen. Zusätzlich setzt der ASG Lookup Service durch, dass Klienten stets das ihnen nächstgelegene Replikat verwenden.

Diese zwei Mechanismen erzeugen gemeinsam einen Feedback-Prozess: Wenn ein Replikat sich den anfragenden Klienten annähert, zieht es zusätzliche Klienten aus dem selben Gebiet an. Dies wiederum erhöht die Attraktivität der Netzwerkregion für das Replikat, was dazu führt, dass es sich weiter in Richtung dieser Region bewegt, und so weiter. Sobald das Replikat in die Gruppe der anfragenden Klienten eintaucht, entsteht ein negativer Feedback-Prozess, welcher die Bewegung des Replikates schnell bremst und zu einer stabilen Konfiguration führt [Her07b].

Der Dienstverteilungs-Algorithmus inspiziert kontinuierlich den Fluss von Nachrichten, der bei einem Replikat eintrifft (*Message Flow*), und trifft lokale Entscheidungen hinsichtlich nötiger Adaptionen. Im Zusammenspiel erzeugen die Instanzen des Algorithmus in den einzelnen Replikaten eine stabile und koordinierte globale Anordnung der Replikate im ASG-Netzwerk [Her07b]. Sobald sich die Anfragemuster der Klienten signifikant ändern, wird auch die Anordnung der Replikate angepasst und konvergiert gegen eine neue globale Konfiguration. Dies wird ohne zusätzliche Kommunikation zwischen den Replikaten realisiert, da diese durch die Nachrichtenflüsse indirekt gekoppelt sind. Der Algorithmus besteht aus drei einfachen Regeln für die unterschiedlichen Adaptionen:

1. **Idle-Regel:** Ein Replikat entfernt sich aus dem System, wenn es weniger als α Anfragen in den letzten m Zeiteinheiten empfangen hat.

2. **Replikationsregel:** Ein Replikat repliziert sich, wenn es einen signifikanten Anfragefluss mit einer durchschnittlichen Pfadlänge größer als ρ empfängt. Das neue

Replikat wird in Richtung dieses Anfrageflusses platziert, das alte verbleibt zunächst in derselben Region.

3. **Migrationsregel:** Ein Replikat migriert zu einem Nachbarknoten, wenn es über diesen einen stabilen Anfragefluss erhält, der *dominant* (größer als die Summe aller übrigen Flüsse) ist.

Der Adaptionsalgorithmus ruft die Regeln in der Reihenfolge auf, in der sie oben aufgeführt sind. Dabei wird die erste Regel, deren Bedingung zutrifft, ausgeführt. Alle weiteren werden übersprungen. Die Idle-Regel stellt einen einfachen Bereinigungsmechanismus dar, der nicht benötigte Replikate beseitigt. Die Replikationsregel übt einen Druck auf das System aus, der bewirkt, dass Replikationen so lange stattfinden, bis jedes Replikat einen Bereich des Netzwerkes von maximal ρ Hops Durchmesser abdeckt. Hierdurch wird eine Gesamtzahl von aktiven Replikaten im System gehalten, die von ρ und vom Durchmesser des Netzwerkes abhängt, und welche effektiv zu einer Aufteilung des Netzwerkes in einzelne Zellen führt. Die Migrationsregel führt dazu, dass Replikate sich zu Stellen bewegen, an denen die Größen aller eintreffenden Anfrage-Flüsse im Gleichgewicht sind. Abbildung 1 zeigt, wie der Gesamt-Nachrichten-Fluss (Zahlen an den Kanten) dadurch reduziert wird, dass das Replikat in Richtung des dominanten Flusses migriert. Das globale Verhalten des Systems (Aufteilung in Zellen und Verringerung der Kommunikation im Netz) ist nicht direkt in den Regeln kodiert. Es ist eine *emergente Eigenschaft* des ASG, die dadurch erzeugt wird, dass eine Anzahl von Replikaten die Regeln anwenden und dabei indirekt interagieren [Her07b].

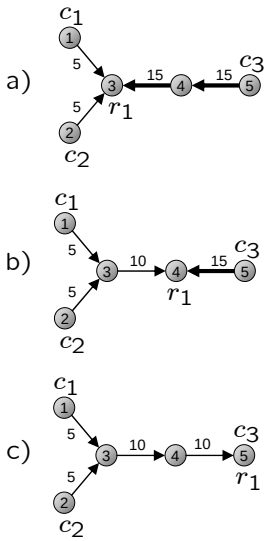

Abbildung 1: Inkrementelle Optimierung der Kommunikationskosten anhand von Anfrageflüssen.

4.2 Service Discovery und Lookup

Die Instanzen des ASG Lookup Service (LS) [HMJ05] laufen verteilt über das gesamte ASG-Netzwerk. Das Netzwerk ist geclustert, und jeder Knoten hat einen *Cluster Head* in seiner direkten Nachbarschaft. Cluster Heads betreiben die Basisdienste (z.B. den LS) im ASG. Jede LS-Instanz speichert Lokations-Informationen zu jedem aktiven Replikat. Aufgrund seiner Verteiltheit müssen Updates dieser Informationen zwischen den Instanzen propagiert werden. Der einfachste Ansatz hierfür wäre Flooding. Da die Bandbreite im drahtlosen ASG-Netzwerk jedoch eine knappe Ressource ist, verwenden wir einen intelligenteren Ansatz, welchen man als *Request-driven Lazy Propagation* bezeichnen kann.

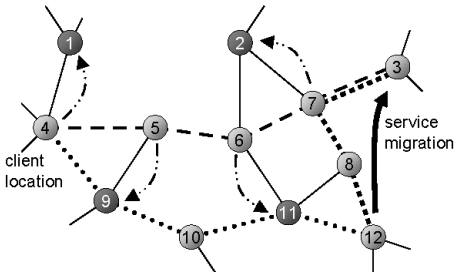

Abbildung 2: Anfrage-getriebene LS-Updates.

Abbildung 2 zeigt den Update-Prozess. Nur diejenigen Nachrichten werden durch das Netzwerk geflutet, die die Erzeugung oder das Entfernen eines Replikates anzeigen. Alle anderen Updates (Migration eines Replikates) werden im *Huckepack-Verfahren* durch normale Antwortnachrichten eines Dienstes transportiert. Ein Knoten, der eine solche Nachricht weiterleitet, inspiziert sie zunächst. Wenn er ein Lokations-Update findet, dann informiert er den LS auf seinem Cluster Head, welcher daraufhin das Update in seine Tabelle eingepflegt. Auf diese Weise werden Updates nur propagiert, wenn der entsprechende Dienst angefragt wird, und das Update verbreitet sich nur in den Regionen des Netzwerkes, aus denen Anfragen kommen. Alle anderen Regionen bleiben zunächst auf dem alten Stand. Allerdings stellt dies kein Problem dar, da ein zweiter Mechanismus dafür sorgt, dass Anfragen bei dem korrekten Replikat eintreffen, obwohl an der Quelle der Anfragen veraltete Informationen vorliegen: Jedes Mal, wenn ein Replikat migriert, hinterlässt es einen *Vorwärts-Zeiger* auf die neue Lokation. Wenn eine Anfrage bei der alten Lokation eintrifft, wird sie weitergeleitet (möglicherweise auch über mehrere Vorwärts-Zeiger). Das Replikat empfängt die Nachricht schlussendlich und generiert eine Antwort, welche dann zusammen mit dem Update an den Sender der Anfrage zurück geschickt wird. Dabei findet das Update wie oben beschrieben statt. Durch diesen Mechanismus wird das Netzwerk sehr unempfindlich gegen Lokationsänderungen und damit robuster. Darüber hinaus ist das Protokoll sehr effizient, da es kaum Lookup-spezifische Nachrichten erzeugt, sondern bestehende Nachrichten als Transportmittel verwendet.

4.3 Datenkonsistenz

Replikate können temporär durch Netzwerkpartitionen getrennt sein. Auch die Tatsache, dass Replikate spontan erzeugt und auch wieder entfernt werden können, sorgt für einen hohen Grad an Dynamik im ASG-System. Daher haben wir einen optimistischen Ansatz zur Konsistenzerhaltung von Replikaten gewählt, der mit einer solchen Dynamik zurecht kommt. Hierbei haben wir das bekannte *Bayou anti-entropy-Protokoll* [TTP+95] erweitert. Das resultierende *Bounded Divergence Group Anti-Entropy Protocol* (BD-GAP) [Her07a] kann einen Abgleich innerhalb einer beliebigen Gruppe von Replikaten durchführen und garantiert *schlussendliche Konsistenz* (evantual consistency). Jedes Replikat kann autonom einen solchen Abgleich starten und sich mit den Replikaten, die es momentan kennt, synchronisieren. Die Information über die aktuell laufenden Replikate erhält es dabei vom Lookup Service. Konflikte, welche aufgrund der gleichzeitigen Ausführung mehrerer Abgleich-Prozesse auftreten, werden automatisch aufgelöst. Das Protokoll wählt die Reihenfolge, in der Replikate Updates austauschen, so, dass die Menge an Daten, wel-

che über das Netzwerk gesendet werden muss, minimiert wird. Des Weiteren begrenzt das Protokoll den Grad, bis zu welchem die Datenspeicher in den einzelnen Replikaten auseinander laufen, so dass die Übertragung des kompletten Datenspeichers, wie er in Bayou notwendig ist, vermieden werden kann.

Das BD-GAP nutzt die Eigenschaften des ASG aus und führt eine stärkere Kopplung zwischen Replikaten ein als das Original-Protokoll, um einen effektiveren Abgleich der Datenspeicher zu erreichen. Dennoch wird eine hohe Verfügbarkeit, wie sie im Bayou-Protokoll realisiert wurde, beibehalten. Das BD-GAP adaptiert sich an die Dynamik im ASG-Netzwerk, indem es bei hoher Dynamik automatisch die Gruppengröße reduziert, bis es schlussendlich wieder bei dem paarweisen Abgleich-Mechanismus angelangt, der in Bayou angewendet wird. In Phasen niedriger Dynamik nutzt BD-GAP dagegen den Umstand aus, dass die meisten Replikate aufeinander zugreifen können, um einen effektiveren Abgleich und damit einen höheren Konsistenzgrad zu erreichen.

4.4 Zusätzliche Ergebnisse

Zusätzlich zu den oben aufgeführten Kernergebnissen haben wir die Middleware MESH*Mdl* entwickelt [HMJ07], welche als Basis für die Implementierung und Evaluierung diente. MESH*Mdl* basiert auf mobilen Agenten und Tuple Spaces, um entkoppelte und asynchrone Kommunikation zu realisieren.

Um zu zeigen, warum und in welcher Weise die vorgeschlagenen Mechanismen selbstorganisierend sind, wurde im Rahmen der Arbeit zusätzlich noch ein *Modell für Selbstorganisierende Softwaresysteme* (SOSS) entwickelt [HWM06]. Dieses Modell dient der Klassifizierung existierender Systeme in solche, die in der Klasse der SOSS liegen und solche, die außerhalb dieser Klasse liegen. Ein solches Klassifizierungs-Werkzeug existierte bislang nicht. Wir glauben, dass eine gezielte Untersuchung existierender Systeme neue Einblicke in die Natur von SOSS bieten wird, welche weit über den Rahmen dieser Arbeit hinaus gehen.

5 Schlussfolgerungen und zukünftige Arbeiten

Im Rahmen dieser Arbeit haben wir die Grundlagen selbstorganisierender Infrastrukturen für Ambient Services gelegt. Die Grundfunktionen, welche wir vorgestellt haben, stellen die Basis für die Entwicklung weiterer Funktionalitäten im AmI-Umfeld dar. Wir haben gezeigt, wie eine selbstorganisierende Dienstverteilung, ein selbstorganisierender Lookup Service und ein adaptives Konsistenzprotokoll modelliert werden können. Weiterhin haben wir diese Konzepte auf der Basis einfacher Middleware-Abstraktionen implementiert und ihre Gültigkeit gezeigt. Unser SOSS-Modell erlaubt uns, präzise zu argumentieren, in welcher Weise die resultierenden Systeme selbstorganisierend sind, was bisher auf dieser formalen Ebene nicht möglich war.

Wir planen, in näherer Zukunft erweiterte Funktionalitäten auf den geschaffenen Grund-

lagen aufzubauen. Die Frage der Sicherheit und Privatsphäre von ASG-Benutzern wurde z.B. bisher nicht betrachtet. Dies wären beides wichtige Aspekte eines kommerziell anwendbaren Systems. Darüber hinaus ist die Frage der Anbindung einzelner ASGs an das Internet und die Föderation von ASGs über Gateways eine weiterführende Aufgabe, welche in Zukunft im Fokus unserer Arbeit stehen wird.

Literatur

[CFLZ05] G. Cabri, L. Ferrari, L. Leonardi und F. Zambonelli. The LAICA project: supporting ambient intelligence via agents and ad-hoc middleware. In *Proceedings of the 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise*, Seiten 39–44, Juni 2005.

[DBS+01] K. Ducatel, M. Bogdanowicz, F. Scapolo, J. Leijten und J.-C. Burgelman. Scenarios for Ambient Intelligence in 2010. Technical Report, The IST Advisory Group (ISTAG), 2001.

[Hel05] Michael Hellenschmidt. Distributed Implementation of a Self-Organizing Appliance Middleware. In Gérard Bailly, Hrsg., *Proceedings of sOc-EUSAI 2005 (Smart Objects Conference)*, Seiten 201–206, 2005.

[Her06] Klaus Herrmann. *Self-Organizing Infrastructures for Ambient Services*. Dissertation, Berlin University of Technology, Juli 2006.

[Her07a] Klaus Herrmann. Group Anti-Entropy – Achieving Eventual Consistency in Mobile Service Environments. In *Proceedings of the 8th IEEE International Conference on Mobile Data Management (MDM'07)*. IEEE Computer Society Press, 2007. (accepted for publication).

[Her07b] Klaus Herrmann. Self-Organizing Replica Placement - A Case Study on Emergence. In *Proceedings of the first IEEE International Conference on Self-Adaptive and Self-Organizing Systems*, Piscataway, NJ, USA, Juli 2007. IEEE Computer Society Press. (accepted for publication).

[HGM04] Klaus Herrmann, Kurt Geihs und Gero Mühl. Ad hoc Service Grid – A Self-Organizing Infrastructure for Mobile Commerce. In *Proceedings of the IFIP TC8 Working Conference on Mobile Information Systems (MOBIS 2004)*. IFIP – International Federation for Information Processing, Springer-Verlag, September 2004.

[HMJ05] Klaus Herrmann, Gero Mühl und Michael A. Jaeger. A Self-Organizing Lookup Service for Dynamic Ambient Services. In *25th International Conference on Distributed Computing Systems (ICDCS 2005)*, Seiten 707–716, Piscataway, NJ, USA, Juni 2005. IEEE Computer Society Press.

[HMJ07] Klaus Herrmann, Gero Mühl und Michael Jaeger. MESHMdl Event Spaces – A Coordination Middleware for Self-Organizing Applications in Ad hoc Networks. *Journal on Pervasive and Mobile Computing – Special Issue on Middleware for Pervasive Computing*, 2007. (accepted for publication).

[HWM06] Klaus Herrmann, Matthias Werner und Gero Mühl. A Methodology for Classifying Self-Organizing Software Systems. In *International Conference on Self-Organization and Autonomous Systems in Computing and Communications (SOAS'2006)*, September 2006.

[IST+05] Valérie Issarny, Daniele Sacchetti, Ferda Tartanoglu, Françoise Sailhan, Rafik Chibout, Nicole Lévy und Angel Talamona. Developing Ambient Intelligence Systems: A Solution based on Web Services. *Automated Software Engineering*, 12(1):101–137, 2005.

[KS03] Rolf Kraemer und Peter Schwander. Bluetooth Based Wireless Internet Applications for Indoor Hot Spots: Experience of a Successful Experiment During CeBIT 2001. *Computer Networks*, 41(3):303–312, 2003.

[OOC+04] Gregory M. P. O'Hare, Michael J. O'Grady, Rem W. Collier, Stephen Keegan, Donal O'Kane, Richard Tynan und David Marsh. Ambient Intelligence Through Agile Agents. In Yang Cai, Hrsg., *Ambient Intelligence for Scientific Discovery*, Jgg. 3345 of *Lecture Notes in Computer Science*, Seiten 286–310. Springer-Verlag, 2004.

[RFPV04] Marcela Rodríguez, Jesus Favela, Alfredo Preciado und Aurora Vizcaíno. An Agent Middleware for Supporting Ambient Intelligence for Healthcare. In *Second Workshop on Agents Applied in Health Care (ECAI 2004)*, August 2004.

[ROPD03] F. Rousseau, J. Oprescu, L.-S. Paun und A. Duda. Omnisphere: A Personal Communication Environment. In *Proceedings of the 36th Hawaii International Conference on System Sciences (HICSS-36 2003)*, 2003.

[TTP+95] D. B. Terry, M. M. Theimer, Karin Petersen, A. J. Demers, M. J. Spreitzer und C. H. Hauser. Managing Update Conflicts in Bayou, a Weakly Connected Replicated Storage System. In *Proceedings of the Fifteenth ACM Symposium on Operating Systems Principles*, Seiten 172–182, New York, NY, USA, 1995. ACM Press.

Klaus Herrmann wurde geboren am 14. Mai 1971 in Rüsselsheim. Er erhielt sein Informatik-Diplom 1998 von der Johann Wolfgang Goethe-Universität in Frankfurt am Main. Anschließend arbeitete er zwei Jahre als wissenschaftlicher Mitarbeiter in Frankfurt, bevor er an die Technische Universität Berlin ging. Dort promovierte er 2006 mit der Arbeit „Self-Organizing Infrastructures for Ambient Services“. Seit September 2006 arbeitet er als wissenschaftlicher Assistent in der Abteilung für Verteilte Systeme am Institut für Parallele und Verteilte Systeme (IPVS) der Universität Stuttgart. Dort betreut er die Forschungsbereiche „Selbstorganisierende Softwaresysteme“ und „Sensornetze“. Im Februar 2007 wurde er für seine Doktorarbeit mit dem Dissertationspreis der GI-Fachgruppe „Kommunikation und Verteilte Systeme“ (KuVS) ausgezeichnet. Seine Forschungs-Interessen umfassen unter anderem Selbstorganisations- und Adaptions-Prozesse in mobilen und spontan vernetzten Systemen sowie in weitläufig verteilten Informationssystemen.