Hochschule für Angewandte Wissenschaften Hamburg
Hamburg University of Applied Sciences

Diplomarbeit

Björn Jensen

Multiagentensystemgestützte Clusteranalyse

Fachbereich Elektrotechnik und Informatik
Department of Electrical Engineering and Computer Science

Björn Jensen

Multiagentensystemgestützte Clusteranalyse

Diplomarbeit eingereicht im Rahmen der Diplomprüfung
im Studiengang Informatik
Studienrichtung Softwaretechnik
am Fachbereich Elektrotechnik und Informatik
der Hochschule für Angewandte Wissenschaften Hamburg

Betreuender Prüfer : Prof. Dr. Kai von Luck
Zweitgutachter : Prof. Dr.-Ing. Martin Hübner

Abgegeben am 3. August 2005

Björn Jensen

Thema der Diplomarbeit

Multiagentensystemgestützte Clusteranalyse

Stichworte

Cluster, Clusteranalyse, Multiagentensysteme, Soziale Software, Verteilte Künstliche Intelligenz, Softwareagenten, Profile, Graphentheorie, verteilte Systeme, Java, J2ME

Kurzzusammenfassung

Bei vielen Arbeitgebern wird projektgebunden gearbeitet. Dies impliziert, dass ein Teil verfügbarer, menschlicher Resourcen zu einem Projektteam (Cluster) zusammengestellt wurde. Der Vorgang der Zusammenstellung erfordert die eingehende Analyse der vorhandenen Resourcen, Auswahl geeigneter Personen und Konfiguration des Teams. Dieser Vorgang ist komplex und zeitaufwendig.

In dieser Diplomarbeit wird eine entsprechende Situation analysiert. Aufbauend auf dem Ergebnis der Analyse wird ein (Multi)agentensystem entworfen und realisiert, welches die oben genannte Aufgabe automatisiert.

Björn Jensen

Title of the paper

Mulitagentsystemsupported Clusteranalysis

Keywords

Cluster, Clusteranalyse, Multiagent systems, Social Software, Distributed Artificial Intelligence, Softwareagents, Profiles, Graphtheorie, Distributed Systems, Java, J2ME

Abstract

Within many companies everydays work ist projectbased. This implies that a part of the available human resource is collocated to a projectteam (cluster). This means a deep analysis of the available human resource, an extract of adequate persons and the configuration of the team. This procedure is complex and time-consuming.

Within this thesis an according scenario is analysed. On the result of the analysis a (multi)agent-system would be designed and realised, which automises the configuration procedure.

Für Mia und Corina

Danksagung

Bevor Sie sich als Leser mit dem zentralen Gegenstand dieser Diplomarbeit beschäftigen, möchte ich das Augenmerk auf etwas richten, was für eine solche Arbeit von essenzieller Bedeutung ist.
So eine Arbeit, die immer mit einem gewissen Maß an zeitlichem Aufwand verbunden ist, stellt nicht nur Forderungen an denjenigen, der die Arbeit erstellt, sie impliziert auch die Gegebenheit von Rahmenbedingungen, die das Erstellen einer solchen Arbeit möglich machen. Es ist immer ein mehr oder weniger großer Personenkreis direkt oder indirekt involviert und diesen Personen möchte ich an dieser Stelle meinen Dank aussprechen.
Vermutlich erreicht mein Dank nicht alle, die es verdienen und mehr, als es eigentlich sollte. Doch jene, die hier benannt werden, haben eine besondere Stellung im Kontext der Erstellung dieser Arbeit inne.

Am Anfang stand eine Idee. Diese Idee konnte ich mit Hilfe von Prof. Dr. Kai von Luck letztendlich zu der vorliegenden Arbeit konkretisieren. Ich weiß seine Unterstützung und seine Geduld zu schätzen.

Peer Hartmann hat mir in Bezug auf die Recherche nach der Quelle bzw. der Wahrheit von Sprichwörtern sehr geholfen.

Und obwohl man sich noch nicht so lange kannte, konnte ich immer auf die Unterstützung des neu gewonnen Familienteils zählen, sei es Sven Wilzog oder Gerd und Monika Grabow.

Auch wenn die Wolken am Horizont manchmal ziemlich düster waren, konnte ich trotz schwerer Zeiten immer auf meine Eltern Jens-Uwe und Renate zählen. Ohne diese beiden (und die fünf anderen) wäre ich nicht, was ich jetzt bin.

Nicht nur meine Eltern, auch mein Bruder Leif hat, wohl mehr unbewusst als bewusst, mir sehr zur Seite gestanden, und sei es nur durch die häufige Frage nach der Thematik, die eine wiederkehrende Auseinandersetzung damit implizierte.

Und was wäre ich ohne Oliver Sander, Peer Hartmann und Charles Ebert als Quelle der Korrektur und der konstruktiven Kritik.

Zu guter Letzt ist es nur durch dem scheinbar unermüdlichen Wesen und der schier grenzenlosen Geduld meiner Frau Corina zu verdanken, dass ich diese Arbeit schreiben konnte. Ihre Kraft hat es mir ermöglicht, der Umstände zum Trotz häufig und lange an dieser Arbeit zu sitzen, auch wenn sie dadurch oft auf mich verzichten musste.

Inhaltsverzeichnis

Tabellenverzeichnis

Abbildungsverzeichnis

1 Einführung

1.1 Einleitung

Als Arbeitnehmer in einem Unternehmen beispielsweise aus dem IT-Bereich arbeitet man zum größten Teil projektgebunden. Wann immer ein neues Projekt startet, wird ein Prozess gestartet, der zum einen schwierig und zum anderen sehr zeitaufwendig ist. Es muss ein Team (ein Cluster) gefunden werden, das dieses Projekt in die Tat umsetzt. Dafür gibt es in der Regel mindestens einen Projektverantwortlichen und je nach Komplexität des Projektes eine Menge von Personen, die in dem Projekt mitarbeiten.

Diese Menge von Personen ist im Allgemeinen eine Untermenge der insgesamt in dem Unternehmen tätigen Personen. Ergo muss die personalverantwortliche Stelle ein entsprechendes Projektteam auswählen und/oder freie Mitarbeiter rekrutieren. Es kommt jedoch auch vor, dass die Konfiguration des Teams dem Projektleiter obliegt. Die Auswahl in Frage kommender Personen erfordert eine eingehende Analyse der Kandidaten, die Zusammenstellung des Teams Zeit, eine Resource, die meist (indirekt) in Geld gemessen wird. Darum erfolgt die Konfiguration häufig nicht in der erforderlichen Güte. Wie kann man dieses Problem lösen?

Eine Möglichkeit ist das sogenannte Outsourcing des Problems: Das Unternehmen übergibt die Lösung des vorhandenen Problems an externes Fachpersonal. In Bezug auf die Teambildung sind dies häufig Zeitarbeitsfirmen oder Personalvermittlungsagenturen.

Eine andere Möglichkeit ist die Delegation der Lösung an einen oder mehrere Computer. Der zentrale Einsatz durch Delegation an einen Computer wurde in der Studienarbeit von Christian Butrynowski (Butrynowski, 2005) untersucht. Der Schwerpunkt dieser Arbeit hingegen liegt auf dem verteiltem Ansatz, also der Delegation an mehrere Computer.

Der geübte Umgang mit Computern kann in dieser Arbeit vorausgesetzt werden, da diese einen immer größeren Stellenwert in unserer Gesellschaft bekommen (siehe Kapitel 1.1.1). Waren diese bis vor ein paar Jahren auf Grund des Verhältnisses von physikalischer Größe zu Leistungsfähigkeit lokal beschränkt, so ist diese Abhängigkeit heutzutage aufgehoben. Der Speicherplatz wird immer kleiner und die Leistungsfähigkeit, die Workstations vor ein

paar Jahren aufgebracht haben, wird durch die moderner, mobiler Endgeräte wie beispielsweise Handys bei weitem übertroffen. Moniert man den Komfort der Eingabemöglichkeit über mehrfachbelegte Tasten eines Nummernblocks, wird dieser Kritikpunkt durch zunehmende Symbiose von sogenannten *Personal Digital Assistants*, kurz PDAs, und Handys in Form von sogenannten SmartPhones entkräftet, da diese meist über eine zwar kleine, dennoch umfangreiche Tastatur verfügen.

Doch die Verbreitung und das Ausmaß moderner Informations- und Kommunikationstechnologie stößt nicht überall auf offene Ohren, wie Kapitel 1.1.2 zeigt. Man wittert Gefahren, die nicht nur den direkten Benutzer betreffen, sondern Auswirkungen auf die gesamte moderne Gesellschaft implizieren.

Im direkten Kontrast dazu steht ein Trend, der in Kapitel 1.1.3 näher betrachtet wird. Der dem Trend zu Grunde liegende Ansatz wird in dieser Arbeit aufgegriffen und weitestgehend modifiziert.

1.1.1 Ein globaler Siegeszug

Es geht in dieser Arbeit nicht nur darum, dass Problem der Clusterkonfiguration beziehungsweise dessen Lösung zu dezentralisieren. Vielmehr wird die Stärke dieser Lösung darin gesehen, dass es sich um ein verteiltes System handelt, da jeder Mitarbeiter über einen in einem Netzwerk kooperierendem Computer verfügt. Diese Computer kommunizieren untereinander. Angesichts der Verbreitung von Computern ist dieses Szenario gar nicht so abwegig.

In der heutigen Gesellschaft ist der Computer als den Menschen unterstützendes Medium kaum noch weg zu denken. So hat eine Technologie, deren Anschaffung vor knapp 25 Jahren für die Allgemeinheit nicht zur Debatte stand, den Siegeszug in unseren Alltag schon längst gewonnen. Ein Blick auf Erhebungen durch das Statistische Bundesamt (Pötzsch u. a., 2003) bestätigt dies[1]. Aus diesen Erhebungen geht hervor, dass in mehr als der Hälfte aller Haushalte Geräte der Informations- und Kommunikationstechnologie (IKT) vorhanden sind. Häufig sind sogar mehrere Geräte vorhanden. Tendenz steigend.

Betrachtet man nur die Haushalte, in denen ein oder mehrere Computer vorhanden sind, stellt man fest, dass die Anzahl derer, die das Internet nutzen, stark ansteigt. Es wird unter anderem für Recherche, der Suche nach Telefonnummern, Zug- oder Straßenverbindungen, der Bestellung von Büchern und CDs, dem Buchen von Reisen und dem Online-Banking genutzt. Der „Net-Economy“ und dem „E-Business“ kommen somit eine zentrale Rolle bei der Diskussion um Wachstum, Beschäftigung oder ganz allgemein Wohlstand zu. Dies gilt nicht nur für Deutschland oder Europa, sondern weltweit.

[1] Bei diesen Erhebungen waren Mehrfachnennungen möglich

Gegenstand der Nachweisung	*Einkommens- und Verbrauchsstichprobe*	*Laufende Wirtschaftsberechnungen*			*IKT-Pilotstudie*
	1998	1999	2000	2001	2002
Erfasste Haushalte (Anzahl)	64056	5693	5827	5850	4962
Hochgerechnete Haushalte (1.000)	34134	34170	34390	34777	37739
			in %		
Handy ohne Internet-Zugang	-	-	-	-	64
Handy mit Internet-Zugang	-	-	-	-	12
ISDN-Anschluss	4	5	8	12	21
DSL/TDSL/ADSL-Anschluss	-	-	-	-	6
Fernsehen mit Satellitenempfangsanlage	29	28	32	32	37
Fernsehen mit Kabelanschluss	54	53	54	54	49
PC stationär	-	43	46	52	55
Notebook, Laptop	-	5	6	6	10
Handheld Computer (Palmtop)	-	-	-	-	3
DVD-Player	-	-	-	-	10
DVD-Laufwerk (integriert im PC)	-	-	-	-	13

Tabelle 1.1: Ausstattung privater Haushalte mit ausgewählten Informations- und Kommunikationstechnologien(IKT) nach unterschiedlichen Erhebungen in Deutschland

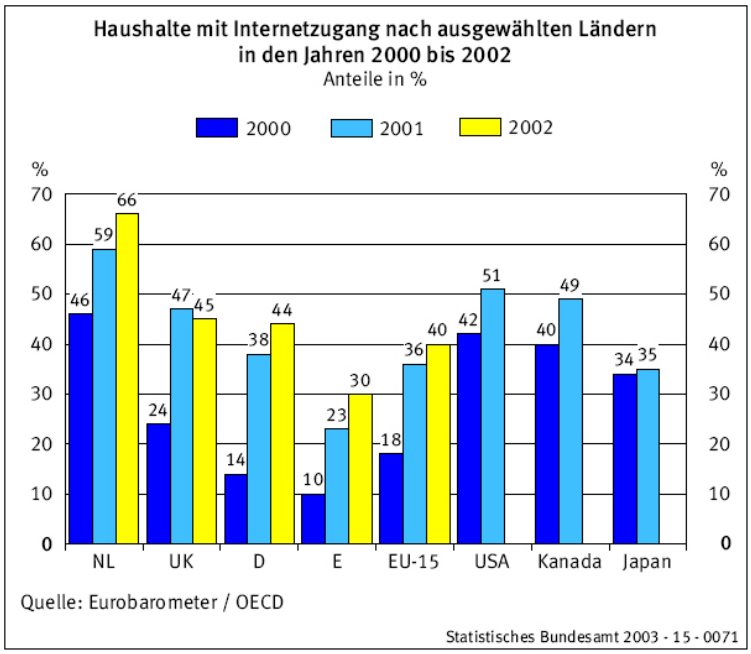

Abbildung 1.1: Übersicht über die prozentuale Verbreitung von Internetzugängen

Allein in Deutschland gehören fast die Hälfte aller Computerbesitzer im Jahre 2002 zur sogenannten Community, jenem Personenkreis, der sich über das Internet verbunden fühlt. Dabei sollte jedoch folgendes berücksichtigt werden: Die Nutzung von Netzwerken beziehungsweise Netzwerkapplikationen jeglicher Art erfolgte bisher der Gestalt, dass alle involvierten Personen bei jeder Aktitvität persönlich dafür Sorge tragen mussten, dass ihren indiviudellen Neigungen und Präferenzen nicht zuwider gehandelt wurde.

1.1.2 Anonymisierung

Die Nutzung von Informations- und Kommunikationstechnologie sowie entsprechender Netzwerke wie beispielsweise dem Internet verbreitet sich zunehmend. Dieser Entwicklung steht man aus psychologischer Sicht sehr kritisch gegenüber. Seit gut 20 Jahren befürchten Soziologen und Psychologen, dass moderne Informations- und Telekommunikationsmedien zur Vereinzelung und Vereinsamung des Menschen führen. Angesichts der Verbreitung globaler Computernetze und der stetig ansteigenden Vielfalt an Diensten gewinnt die Vernetzung-Kritik eine neue Brisanz. Man fürchtet, dass der Benutzer jener Technologien und Dienste sich im Sog der Virtuellen Realitäten befindet. Diese Realitäten stellen ein perfektes Konkurrenzmodell zur Wirklichkeit dar. Realitätsflucht und Rückzug aus den realen sozialen Beziehungen sind die Folge.

Dies sind jedoch „nur“ die Folgen. Schlimmer noch wiegt die Tatsache, dass genau dieser Rückzug aus verbindlichen zwischenmenschlichen Beziehungen zugunsten austauschbarer, flüchtiger TeleKontakte dem „(post)modernen Menschen“ und seiner egozentrischen, auf Abwechslung und Unverbindlichkeit ausgerichteten Lebensweise und Wertestruktur entgegenzukommen scheint. Die zwischenmenschliche Kommunikation wird durch die Mensch-Maschine-Kommunikation substituiert.

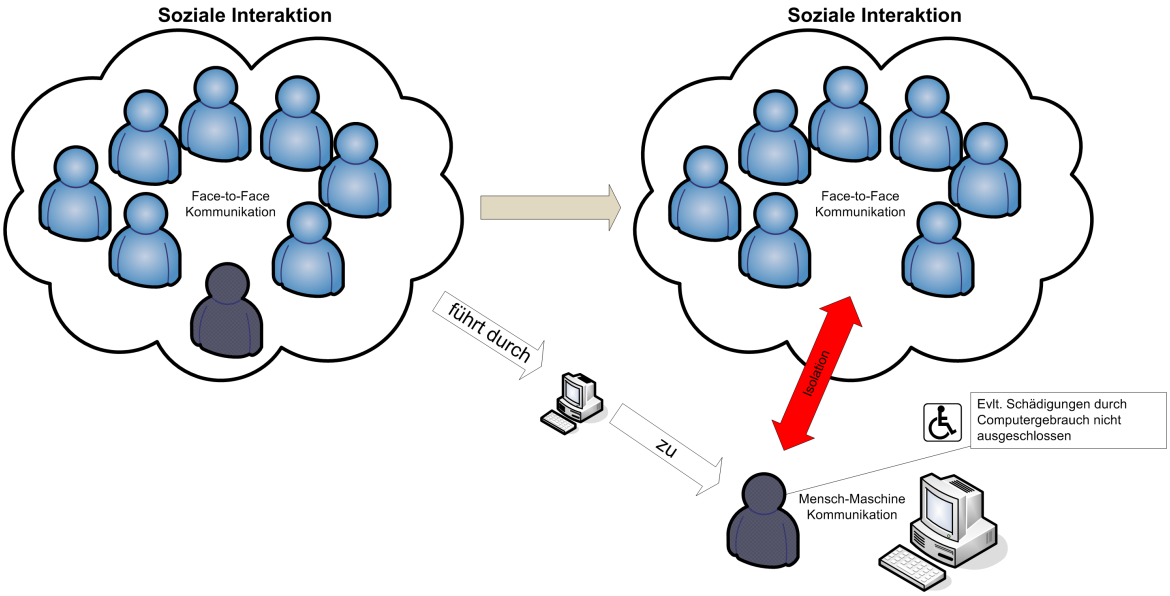

Abbildung 1.2: Vernetzungskritik

Durch diese Substitution entsteht ein Mangel an zwischenmenschlichem Austausch, der Entfremdungs- und Einsamkeitsgefühle auslösen kann und den Menschen wichtiger sozialer Lern- und Entwicklungsmöglichkeiten beraubt.

Der „moderne“ Mensch technisiert zunehmend durch die intensive Auseinandersetzung mit dem Computer. Dies beruht auf der Anpassung des menschlichen Individuums an die formallogische Maschinenwelt, welche durch Steuerbarkeit und Kontrolle, Regelhaftigkeit und Eindeutigkeit, sowie Zweck-Mittel-Rationalität geprägt ist.

Die freiwillige Isolation folgt letztendlich daraus, dass das Streben nach der Beherrschung der Maschine Macht und Überlegenheit suggeriert. Der technisierte Mensch wird schließlich den Umgang mit Maschinen dem sozialen Zusammensein vorziehen, weil letzteres mit Unkontrollierbarkeit, Vieldeutigkeit, Komplexität und Ambivalenz verbunden ist.

Erschreckend ist daran, dass diese Isolation und der Rückgang zwischenmenschlicher Kommunikation unbewusst verläuft. Deshalb können auch keine „Widerstandskräfte gegen die wachsende Selbst- und Fremdisolierung und die steigende Einsamkeit in unserer Gesellschaft mobilisiert werden.“

Auch die Gegendarstellung der Community, dass man sehr wohl zwischenmenschliche Kommunikation (zum Beispiel durch Chat-Räume) pflege, ist in Augen der Kritiker nicht richtig. Denn die Form der Face-to-Face-Kommunikation ist nur quasi-zwischenmenschlich, da die Kontaktaufnahme gänzlich anonym und unverbindlich ist. Durch das vollständige Fehlen nonverbaler Signale wie Stimme und Stimmung, Geruch und Atmosphäre kann die urspüngliche Form der zwischenmenschlichen Kommunikation nicht erreicht werden. Die Anpassung des Menschen an den Computer führt zu Formalisierung, Entemotionalisierung, Entzeitlichung, Enträumlichung, wenn nicht sogar zu Entmenschlichung.

Auch wenn zunehmend andere Wissenschaften sich auf Seite der Kritiker schlagen, steht den Spekulationen und Befürchtungen nur wenig Empirie gegenüber. Ad absurdum wird die Diskussion erst recht dadurch geführt, dass viele der Vernetzungskritiker sich mittlerweile für den Universal Access, den kostengünstigen Netzzugang für alle, engagieren (Döring, 1996a,b).

1.1.3 Social Software

Der Mensch ist ein soziales Wesen
Meg Hourihan
(Sixtus, 2005)

In Hinblick auf den Gegenstand dieser Arbeit steht auf der einen Seite also das Problem, Menschen zu gruppieren, auf der anderen Seite der Lösungsansatz, Computer dezentral dazu einsetzen zu wollen. Rückblickend auf Kapitel 1.1.2 scheinen somit Problem und Lösungsansatz sich gegenseitig auszuschließen, oder etwa nicht?

Seit einiger Zeit macht ein neuer Trend von sich reden: Social Software. Unter Social Software (dt. Soziale Software) bezeichnet man (Software-)Systeme, die folgende Dinge unterstützen:

- Menschliche Kommunikation,
- Interaktion und
- Zusammenarbeit.

Soziale Software hat sich um 2002 im Zusammenhang mit damals neuen Anwendungen wie Wiki und Weblogs etabliert. Doch schließt der Begriff auch Dienste und Anwendungen mit ein, die deutlich älter sind, so lange die oben genannten Dinge unterstützt werden.

Soziale Netzwerke und Communitites werden durch alle sozialen Systeme in Aufbau und Pflege unterstützt. Diese Systeme funktionieren weitestgehend durch Selbstorganisation (Wikipedia 2005).

Der Begriff der sozialen Software steht seit her jedoch häufig für Webanwendungen, da diese auf der Viele-zu-Viele-Kommunikation aufbauen. Doch muss man sich vor Augen halten, dass es sich hierbei eher um indirekte als um direkte Mensch-zu-Mensch-Kommunikation handelt.

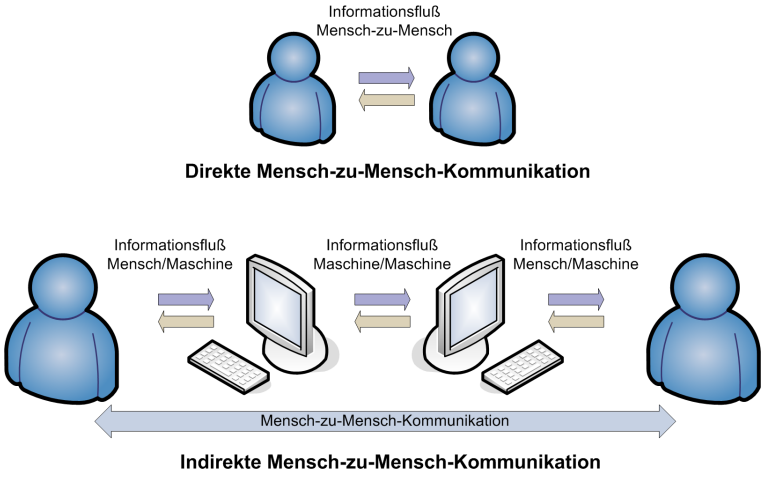

Abbildung 1.3: Indirekte Mensch-zu-Mensch-Kommunikation

Der zunehmende Erfolg sozialer Software in Bezug auf das Web liegt laut Joi Ito, Chairman von Six Apart Japan, daran, dass es das Internet in seiner ursprünglichen Form nicht mehr gibt. Viel mehr erlebt man nun die Metamorphose in eine neue Form des World Wide Web. So gibt es im Grunde zwei parallele Welten innerhalb des Netzes. Die eine fokusiert den kompletten kommerziellen Bereich, die andere kann alles andere sein. Die Gemeinsamkeit ist jedoch auch hier das Auftreten „anonymer, gesichts- und geschichtsloser CyberWesen“. Doch der Hauptunterschied zum World Wide Web der Stunde null ist der, dass der Personenkreis, bei dem der Computer einen relativ zentralen Punkt im Leben darstellt, nicht mehr alleiniger Benutzer des Netzes ist, da sich ihm der „normale“ Mensch zur Seite stellt. Das Web der zweiten Generation wird zunehmend dazu genutzt, Beziehungen aufzubauen und zu pflegen. Dabei beschränkt sich der Kontakt zunehmend nicht mehr alleinig auf die Grenzen des Internets - mit Erfolg (Möller, 2005; Sixtus, 2005; Langham, 2005).

1.2 Problemstellung

Isn't our goal rather to interact with information, to communicate and to collaborate with people?

Norbert Streitz, Paddy Nixon
(Streitz und Nixon, 2005)

Gegenstand des in dieser Arbeit behandelten Problems ist die dezentralisierte Clusteranalyse. Hierzu ist ein entsprechendes Softwaresysten notwendig. Um was für ein System es sich dabei handelt, soll im Folgenden geklärt werden.

Def.: Ein System ist im folgendem synonym für einen Computer oder eine Applikation.

Das System muss weitestgehend autark sein und im Sinne seines Benutzers handeln, wenn der Benutzer des selbigen nur minimal gefordert werden soll. Nur bei Bedarf fordert das System Anweisungen oder Informationen zu einem bestimmen Handeln direkt von seinem Benutzer an.

Es gibt für jeden Benutzer ein solches System. Diese System arbeiten in einem Netzwerk zusammen. Somit agieren Systeme in einer Art künstlicher Gesellschaft. Innerhalb dieser Gesellschaft können diese Systeme interagieren und bei Bedarf kommunizieren sowie Informationen weitergeben. Das führt dazu, dass die Mensch-zu-Maschine-Kommunikation transparent wird und die Form einer indirekte Mensch-zu-Mensch-Kommunikation annimmt, bei der das System weitestgehend in den Hintergrund „gedrängt" wird.

Die zwischenmaschinelle Kommunikation gekoppelt mit dem Informationsaustausch der inzidenten[2] Benutzer soll in dieser Arbeit dahingehend genutzt werden, dass man sich diese zur optimalen Zusammenstellung von Clustern zu Nutze machen kann.

Der Vorgang dieser Zusammenstellung ist die Clusteranalyse, ein weiteres Problem, das in dieser Arbeit behandelt wird. Dabei liegt das Problem jedoch weniger in der Bildung von Gruppen, sondern viel mehr darin, welche Kriterien Basis der Gruppenbildung sind. Die in dieser Arbeit behandelte Gruppenkonfiguration ist sehr brisant, da es sich bei den Elementen einer Gruppe nicht um Maschinen, sondern um Menschen handelt. Eine falsche Konfiguration kann schwere Auswirkungen haben.

[2] Inzidenz bedeutet in diesem Fall so viel wie zusammengehörig, zugeordnet. In der Graphentheorie beschreibt das Wort Inzidenz das Verhältnis einer Ecke zu allen an dieser Ecke ankommenden beziehungsweise von dieser Ecke abgehenden Kanten im gerichteten Fall. Im ungerichteten Fall gilt diese Eigenschaft analog dazu.

Die den Benutzer in der künstlichen Gesellschaft repräsentierenden Systeme, die weitestgehend im Sinne ihres Benutzers handeln, agieren im Sinne eines Agenten.

1.3 Zielsetzung

Aufbauend auf der Problemstellung soll prototypisch eine soziale Software zur Clusteranalyse entworfen und realisiert werden, welche ihren Benutzer minimal fordert und in einer gegebenen Aufgabe maximal unterstützt. Das System soll zunächst auf einer größereren Rechnerarchitektur wie etwa einem Personalcomputer laufen und dann auf kleinere Systeme wie PDAs oder SmartPhones portiert werden.

1.4 Beispielszenario

Dem Beispielszenario dieser Arbeit liegt das Beispielszenario der Studienarbeit von Christian Butrynowski (Butrynowski, 2005) zu Grunde. Darin geht es um den Gast eines Ferienclubs[3]. Diesem Gast steht unter anderem ein eigener Computer für die Dauer seines Aufenthaltes in dem Club zur Verfügung.

Neben den bekannten Nutzungsmöglichkeiten[4] verfügt das bereitgestellte System über ein besonderes Personal Information Management System (PIM). Dieses erfordert von dem Gast die Eingabe gewisser Profildaten. Ebenfalls informiert es den Gast über alle Aktivitäten des Clubs und agiert bei Interesse des inzidenten Benutzers in angemessener Art und Weise.

Will der Gast beispielsweise gemeinsam mit anderen Gästen speisen, sorgt das System für einen Sitzplatz in möglichst angenehmer Umgebung sowohl vom Ambiente als auch in Bezug auf die Personen, die sich in unmittelbarer Nähe des Gastes befinden.

Wenn der Gast etwas unternehmen möchte und Personen sucht, die ebenfalls Interesse an dieser Unternehmung haben, so kann er die Anfrage dem PIM übergeben. Dieses wird dann prüfen, ob es einen entsprechenden Personenkreis gibt. Das PIM in dieser Ausprägung ist zusammen mit einer Agentenkomponente jenes System, das sowohl in der Problemstellung als auch in der Zielsetzung erwähnt wurde.

[3] Dieser Ferienclub existiert virtuell an der Hochschule für Angewandte Wissenschaften Hamburg (im folgenden HAW Hamburg) und bot/bietet Raum für diverse Themen und Szenarien im Kontext wissenschaftlicher Arbeiten.

[4] Diese Möglichkeiten unterliegen natürlich den Regeln des Ferienclubs und der System-Administratoren und sind entsprechend eingeschränkt. Der Umfang der Einschränkung ist jedoch erst einmal für diese Arbeit bedeutungslos und wird deshalb außer Acht gelassen.

Da die Menge der möglichen Dienste, die das PIM seinem Benutzer zur Verfügung stellen kann, den Rahmen dieser Arbeit sprengen würde, wird in dieser Arbeit beispielhaft die Teilnahme an einem gemeinschaftlichem Essen als Unternehmung genannt und untersucht. Cluster sind in diesem Kontext all jene Personen, die gemeinsam an einem Tisch sitzen.

1.5 Gliederung

Der Aufbau dieser Arbeit kann an dieser Stelle (grob) betrachtet werden. So ist diese Arbeit in drei große Bereiche unterteilt. Diese Bereiche bauen auf dem jeweils vorangegangenen Bereich auf und sollten möglichst nicht unabhängig von einander gelesen werden.

Der erste Teil endet mit dieser Gliederung. Man wurde in diesem Kapitel allgemein in die behandelte Problematik eingeführt. Es wurden die technische Entwicklung und deren Auswirkung auf unsere Gesellschaft aus unterschiedlichen Blickwinkeln betrachtet. Ebenso wurde der Trend erläutert, den Computer so klein und leistungsfähig wie möglich zu machen. Ausgehend auf diesem Wissen wurde die vorgestellte Problematik konkretisiert und das Ziel dieser Arbeit diesbezüglich geklärt. Dieses Ziel wurde dann an einem Beispielszenario festgemacht.

Teil 2 stellt den Schwerpunkt der Arbeit dar. Er besteht aus den Kapiteln 2 und 3. Dieser Teil zeigt den Weg vom Problem zur Lösung auf. Dabei besteht dieser Weg aus 3 Phasen:

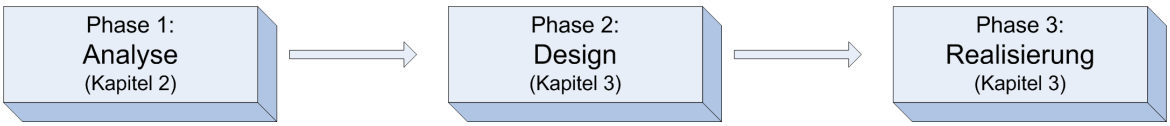

Abbildung 1.4: Die Schwerpunkte der Arbeit

Die Analyse (Kapitel 2) sollen alle notwendigen Anforderungen gezeigt werden. Sie klärt die Begrifflichkeiten, mit denen man in dieser Arbeit konfrontiert wird. Es werden Verfahren zur Gewinnung und Verarbeitung notwendiger Informationen vorgestellt und man beleuchtet das System, dessen Entwicklung Ziel der Arbeit ist, und seinen Kontext im Detail.

Das Design (Kapitel 3) zeigt auf, wie die Ergebnisse der Analyse praktisch umgesetzt werden können und stellt die Architektur des Systems dar. Ebenfalls wird in diesem Kapitel auf die Realisierung eingegangen. Darin werden die Designvorschläge umgesetzt und unter anderem auch Probleme und deren Lösung beziehungsweise deren Lösungsansätze vorgestellt.

Der letzte Teil besteht aus Kapitel 4. Es fasst die gewonnenen Erkenntnisse zusammen, vergleicht diese mit denen aus (Butrynowski, 2005) und gibt einen Ausblick auf mögliche Einsatzgebiete und Visionen.

2 Analyse

In diesem Kapitel beschäftigt man sich eingehend mit der Problemstellung aus Kapitel 1.2 und dem Beispielszenario aus Kapitel 1.4 mit Hinblick auf das Ziel dieser Arbeit (siehe Kapitel 1.3). Der Gegenstand dieser drei Kapitel wird eingehend untersucht und es wird geklärt, welche Aspekte beim Design und später bei der Realisierung berücksichtigt werden müssen. Dabei gliedert sich dieses Kapitel in mehrere Bereiche.

Das für einen Benutzer agierende Agentensystem sollte für den Benutzer bei Bedarf ad hoc zugänglich sein. Diese Form der Omnipräsenz genießt eine immer größere Aufmerksamkeit und soll in Kapitel 2.1 erläutert werden.

Im Gegensatz zu Kapitel 2.1, in dem ein amerikanische Forschungsschwerpunkt vorgestellt wird, stellt Kapitel 2.2 ein artverwandten, europäischen Forschungsschwerpunkt vor.

Nach diesen notwendigerweise eher allgemeinen Betrachtungsweisen und Erläuterungen wird die Analyse nun konkreter. In dem Beispielszenario werden diverse Anforderungen deutlich, die sich in mehrere Gruppen unterteilen lassen. Dabei widmet sich Kapitel 2.3 den Anforderungen, die die Informatik grundsätzlich an zu entwickelnde Applikationen stellt. Kapitel 2.4 untersucht die datenschutzrechtlichen Aspekte. Jedoch ist diese Untersuchung relativ oberflächlich, da eine tiefergehende Untersuchung den Rahmen einer Diplomarbeit sprengen würde. Bei Kapitel 2.5 wird der Fokus auf die Protagonisten des Beispielszenarios gelegt. Es werden Rollen und Anwendungsfälle eruiert und erklärt.

Aufbauend auf den Anforderungen soll in Kapitel 2.6 dargelegt werden, wie eine Applikation den inzidenten Benutzer in Form des eingangs erwähnten Agenten in einer künstlichen Gesellschaft vertreten kann bzw. was die Grundlage der in dem Beispielszenario vorgestellten Interaktion ist: das Benutzerprofil.

Welche Formen es gibt, die dargelegte Form des Benutzerprofils mit Informationen zu füllen, soll in Kapitel 2.7 geklärt werden.

Mit Blick auf die Tatsache, dass in der Einleitung die Delegation der Problemlösung auf mehrere Systeme erwähnt wird, behandelt Kapitel 2.8 die sogenannten „Verteilten Systeme“.

Abschließend wird in Kapitel 2.9 der Begriff des Agenten auf das in der Künstlichen Intelligenz (KI) vorhandene Analogon überführt. Dieses wird aus Sicht der Softwareentwicklung in allen für diese Arbeit notwendigen Facetten untersucht.

2.1 Ubiquitär vs. pervasiv

Hält man sich das Ziel und die Vision der Arbeit vor Augen, dass die Applikation ihren Benutzer maximal unterstützen und minimal fordern soll, bedeutet dies folgendes: Die Applikation und ihre Dienste müssen ad hoc, also auf Abruf ohne Verzögerung verfügbar sein. Es bedeutet außerdem, dass der Benutzer lokal diese Applikation nutzen kann und seinen Standort dafür nicht ändern muss, was dazu führt, dass die zu Grunde liegende Architektur portabel („mobil“) sein muss. Bei der Arbeit beziehungsweise Beschäftigung mit mobilen Geräten und in Hinblick auf die Forderung nach Omnipräsenz stößt man schnell auf das Stichwort Ubiquitious - allgegenwärtiges - Computing. Doch wo unter der Anwendung der mobilen Technik meist das Surfen im Internet verstanden wird, geht das Ubiquitious Computing weit darüber hinaus.

Def.: Unter den beiden oft äquivalent gebrauchten Begriffen Pervasive Computing und Ubiquitious Computing wird die Allgegenwärtigkeit von Informationsverarbeitung und der damit einhergehend der jederzeitige Zugriff auf Daten von beliebiger Stelle aus verstanden (Mattern 2005).

Ubiquitäres Computing hat wie die Virtuelle Realität das Schaffen eines Vorteils und Nutzens des Menschen durch Computertechnik zum Ziel. Jedoch verhalten sich ubiquitäres Computing und virtuelle Realität oppositionell zueinander. Bei der virtuellen Realität wird das oben genannte Ziel durch das Schaffen einer künstliche Realität, in der sich er Nutzer aufhält, erreicht.

Beim ubiquitären Computing erreicht man das Ziel dadurch, dass man die eingesetzte Technik - zum Beispiel ubiquitäre Geräte - überall und jederzeit verfügbar macht. Die Lebensqualität des Nutzers soll sich durch diese technische Unterstützung bei der täglichen Arbeit und im täglichen Leben verbessern. Dem Nutzer sollen idealerweise Arbeit abgenommen und neue Möglichkeiten, die ohne ubiquitäre Geräte nicht möglich wären, eröffnet werden. Dies gilt insbesondere unter Berücksichtigung von Aspekten der Kommunikation und des Wissensbereiches.

Ubiquitäre Technik kann also als Menge der vorhandenen (eingesetzten) Technik und der bereitgestellten Dienste verstanden werden. Dabei können ubiquitäre Geräte als solche verstanden werden, die über eine integrierte Informationstechnologie verfügen, welche jederzeit mit anderen mobilen Geräten (z.B. Handy, PDA, Laptop, etc.) und/oder stationären IT-Systemen mittels kabelloser Kommunikationstechnologien (spontan) kommunizieren können.

Bereitgestellte Dienste können der Zugriff auf das Internet, Intranet, das eigene Mobilgerät, fremde Mobilgeräte oder aber ein Server sein.

In dieser Arbeit werden die Begriffe Ubiquitious und Pervasive Computing synonym verwendet, jedoch gibt es zwischen diesen beiden Begriffen einen deutlichen Unterschied: Unter ubiquitärem Computing versteht man streng genommen die Integration von ubiquitärer Technik in Alltagsgegenstände. Somit verschwindet die eingesetzte Technik in den Hintergrund und wird nicht mehr bewusst wahrgenommen. Das bedeutet, dass die Alltagsgegenstände nun über alle wesentlichen Merkmale der ubiquitären Geräte verfügen.

Pervasives Computing hat seinen Schwerpunkt in der Anwendung und Nutzung in e-Commerce-Szenarien und webbasierten Geschäftsprozessen. Der Begriff des ubiquitären Computing steht in dieser Arbeit für beide Begriffe, da das ursprüngliche ubiquitäre Computing in der gedachten Form noch nicht verfügbar und das pervasive Computing zu speziell ist (Kollakowski, 2004; Babic, 2003).

2.2 Ambient Intelligence

Nun geht es in dieser Arbeit darum, neben einer quasi permanent verfügbaren Applikation auch eine soziale Software zu entwickeln. In Abschnitt 1.1.3 wurden mehrere Bedingungen genannt, die eine Applikation erfüllen muss, um als soziale Software bezeichnet werden zu können, nämlich die Unterstützung in den Bereichen

- Menschliche Kommunikation,
- Interaktion und
- Zusammenarbeit.

Dies betrifft - abstrakt gesehen - sogar unsere Umwelt. Das sich damit befassende Forschung steht unter der Überschrift „Ambient Intelligence“.

Ambient Intelligence steht im engen Zusammenhang mit dem europäischem Forschungsprogramm *Information Society Technologies* und ist verwandt mit dem Ubiquitious Computing (siehe Kapitel 2.1). Der Unterschied besteht jedoch darin, dass der Forschungsbereich des Ubiquitious Computing den Schwerpunkt auf eine stärkere Hardware-Orientierung legt.

Sowohl Ambient Intelligence als auch Ubiquitious Computing werden unter dem Begriff *Next Generation Media* zusammengefasst. Ein spezieller Forschungsbereich ist hier das so genannte *Wearable Computing*. Dabei wird die Unterstützung durch IT direkt in die Arbeitskleidung integriert.

Das Ziel der Forschungsanstrengungen besteht wie im Ubiquitious Computing darin, den Alltag des Menschen zu verbessern. Man will dies durch massive Vernetzung von Sensoren,

Funkmodulen und Computerprozessen erreichen[1]. Sogenannte Sensornetzwerke können zahllose Überwachungsaufgaben übernehmen, sei es im Feuerschutz, der Erdbebenfrühwarnung oder der Verkehrsflusskontrolle. Blinde Akzeptanz sollte man jedoch auch hier nicht auf Grund der Vorteile walten lassen. Ambient Intelligence ist auch mit Vorsicht zu genießen, denkt man in Richtung Kontrolle oder den „gläsernen Menschen“.

Letztendlich steht hinter Ambient Computing eine Vision: Man möchte, dass die Nutzung von Computern den Benutzer ebenso wenig fordert wie andere alltägliche Aktivitäten wie essen, gehen, lesen oder schlafen (Wikipedia 2005).

2.3 Anforderungen aus der Informatik

Neben den kontextspezifischen Anforderungen an die zu entwickelnde Software gibt es eine Reihe allgemeiner Anforderungen, die erfüllt werden sollten. Diese generellen Anforderungen haben sich über Jahre herauskristallisiert und erstrecken sich auf alle Bereiche der Informatik. Die Einhaltung dieser Anforderungen hat sich als überaus sinnvoll erwiesen. Dabei sind diese Anforderungen nicht eigenständig, sondern als Anforderungskomplex zu verstehen, da die einzelne Anforderungen in Wechselwirkung zueinander stehen.

Modularität

Unter Modularität versteht man die Unterteilung eines Themenkomplexes in mehrere kleinere, eigenständige Teile. Diese Teile bezeichnet man als Module. Dabei müssen Absprachen über Schnittstellen unbedingt eingehalten werden, da die Modularität ansonsten nicht mehr gewährleistet werden kann. Das bedeutet, dass Entwickler sich vor der Implementation eines Moduls über den Umfang von Schnittstellen verständigen, über die auf das Modul zugegriffen werden kann beziehungsweise dieses auf andere zugreift. Die Kommunikation zwischen Modulen erfolgt nur über diese Schnittstellen.

Mit dieser Aufteilung umgeht man das Problem des Austausches von Teilbereichen, denn der Austausch von Modulen kann ohne Änderung oder Anpassung anderer Module erfolgen. Dadurch erhöht sich zusätzlich die Wartbarkeit des Gesamtsystems. Die Eigenständigkeit der Teilmodule fördert die Wiederverwendbarkeit.

[1] Erste Anwendungsgebiete sind z.B. das intelligente Haus, in dem sämtliche Einrichtungen (Wärme, Küchenmaschinen, Rollläden, etc.) sich mit (mobilen) Computern (PDAs) von überall her bedienen lassen und sich adaptiv auf die Bedürfnisse der Bewohner einstellen, sowie die effizientere Nutzung der Verkehrsinfrastruktur.

Wiederverwendbarkeit

Wiederverwendbarkeit bedeutet den Einsatz von Komponenten eines Systems in ein anderes Produkt. Durch Anstreben der Wiederverwendbarkeit lässt sich unter anderem die Entwicklungszeit neuer Produkte erheblich verkürzen. Neben der reinen Entwicklungszeit verkürzt sich auch die Zeit, die mit dem Testen der Software verbracht wird, denn einmal getestete Komponenten müssen nicht erneut getestet werden.

Wartbarkeit und Erweiterbarkeit

Im Grunde ist ein System nie fertiggestellt, denn es bleibt nicht aus, dass mit der Zeit neue Anforderungen an das System gestellt werden. Das System muss angepasst, Fehler müssen behoben, neue Funktionen hinzugefügt und alte entfernt werden.

Dieser Umstand macht es notwendig, dass schnell und unkompliziert auf neue Anforderungen reagiert werden kann. So muss das Hinzufügen und Entfernen von Funktionalitäten sowie das Beheben von Fehlern jederzeit und auch ohne Expertenwissen möglich sein.

Portabilität

Dies ist die Fähigkeit, ein System von einer Umgebung auf eine andere zu übertragen. Im Bereich des Software Engineering versteht man darunter meistens die Übertragbarkeit einer Applikation von einem Betriebssystem auf ein anderes beziehungsweise auf unterschiedliche Hardwareplattformen. Dadurch erreicht man Unabhängigkeit gegenüber unterschiedlichen Hard- und Softwareherstellern. Dabei wird angestrebt, diese Portierung möglichst ohne viel Aufwand und/oder Modifikation zu vollziehen.

2.4 Anforderungen aus dem Bereich des Datenschutzes

Die zu entwickelnde Anwendung sammelt Daten eines Nutzers und speichert diese in gewisser Art und Weise. Da jedoch auch personenbezogene Daten darunter sein können, ist die Handhabe eben jener Daten mit äußerster Sorgfalt zu tätigen.

So dürfen ohne Einwilligung des Nutzers keine personenspezifischen Daten gesammelt werden. Außerdem muss der Nutzer Einfluss nehmen können, welche Informationen unter welchen Bedingungen dargelegt werden.

Aus informationstechnischer Sicht gibt es den Ansatz einer Plattform, die diese Aspekte berücksichtigen soll: das Platform for Privacy Preferences Project (siehe Abschnitt 2.7.4).

Diese kurze Abhandlung erhebt keinen Anspruch auf Vollständigkeit. Man könnte an dieser Stelle noch detailierter auf die datenschutzrechtlichen Bestimmungen eingehen, doch ist

der Datenschutz nicht Gegenstand dieser Arbeit und soll von daher nur am Rande erwähnt werden.

2.5 Anforderungen aus dem Beispielszenario

2.5.1 Rollen

Bei der Analyse des Beispielszenarios (Kapitel 1.4) stellt man fest, dass die involvierten Akteure in zwei Gruppen unterteilt werden können. Diese beiden Gruppen sind durch ihre Eigenschaften definiert. Dabei werden diese Gruppen auch als Rollen bezeichnet. In seiner Arbeit hat Christian Butrynowski (Butrynowski, 2005) dieselben Rollen erkannt.

Gast

Der Gast besucht den Ferienclub. Ihm steht ein System zur Verfügung, welches seinem Benutzer ad hoc diverse Möglichkeiten eröffnet. So kann der Gast über das System seine Teilnahme an verschiedenen Aktivitäten regeln (lassen). Voraussetzung zur optimalen Funktionalität ist die Eingabe und Wartung eines Profils, welches möglichst genau Interessen und Neigungen des Benutzers widerspiegelt. In Fall des Beispielszenarios wird dem Gast die Teilnahme an einem gemeinschaftlichem Essen offeriert. Nimmt der Gast die Einladung an, so kann dieser sich durch das PIM über seinen aktuellen Sitzplatz informieren. Dieser kann sich jedoch im Laufe der Zeit ändern.

Administrator

Der Administrator ist Mitglied des Personals des Ferienclubs. Er ist nicht für die Systemadministration zuständig, sondern hat direkten Einfluss auf die Möglichkeiten an Aktivitäten, die dem Gast angeboten werden. In diesem Fall handelt es sich um das gemeinschaftliche Essen. Ebenso hat der Administrator direkten Einfluss auf den Umfang des Profils, dass der Gast im Rahmen seiner Möglichkeiten eingeben und warten sollte.

Weitere Rollen

Neben diesen beiden Rollen sind noch weitere Rollen denkbar. Einige weitere wurden in diversen Arbeiten an der Hochschule für Angewandte Wissenschaften Hamburg im Kontext des Ferienclubs betrachtet (unter anderem von André Lüpke (Lüpke, 2004) oder Marco Bresch (Bresch, 2004)). Jedoch sollen diese hier nur kurz erwähnt und nicht weiter verfolgt werden, da sie für diese Arbeit nicht relevant genug sind. Diese Aufzählung erhebt keinen Anspruch auf Vollständigkeit.

- Rezeption
- Animateur
- Dienstanbieter bzw. Kursleiter
- Systemadministrator
- Interessent
- Animateur
- etc.
- PDA-Administrator
- Reisebüro
- Drittanbieter
- Clubmanager
- Abrechungsssystem
- Drittanbieter

2.5.2 Anwendungsfälle

Betrachtet man das Beispielszenario (Kapitel 1.4) auf die Aktionsmöglichkeiten der dedizierten Rollen hin, so werden diverse Anwendungsfälle klar. Diese Anwendungsfälle sollen im folgenden kurz beschrieben werden. Jedoch werden die Anwendungsfälle des Administrators in dieser Arbeit nicht berücksichtigt, da dieser mit der eigentlichen Applikation nicht viel zu tun hat. Der Vollständigkeit halber werden diese Anwendungsflälle dennoch vorgestellt.

2.5.2.1 Anwendungsfälle des Administrators

Analog zu anderen Administratoren in der IT, hat der in dem Beispielszenario genannte Administrator als priviligierter Benutzer im Vergleich zum gemeinen Benutzer - dem Gast - mehr Rechte und entsprechend mehr Möglichkeiten der Aktion. Allgemein kann der Umfang der Möglichkeiten beliebig sein, bezogen auf das Beispielszenario stellen die folgenden Aktionen das notwendige Gerüst einer eingehenden, tieferen Betrachtung dar.

Profilvorlage anlegen

Die im Beispielszenario genannte Applikation agiert hauptsächlich mit Informationen, die der jeweilige Benutzer von sich Preis gibt. Damit jedoch diese Informationen miteinander verglichen werden können, muss quasi ein Standard definiert werden. Dieser Standard existiert in Form der Profilvorlage, welche durch den Administrator angelegt werden muss.

Profilvorlage bearbeiten

Stellt sich heraus, dass der Umfang der Profilvorlage zu mächtig oder zu gering ist, muss diese durch den Administrator entsprechend angepasst werden. Das ist unter anderem dann der Fall, wenn eine Veranstaltung angeboten wird, die nicht durch das vorhandene Profil abgedeckt wird. Als Beispiel sei hier das Angebot einer musikalischen Veranstaltung, wobei in dem Profil musikalische Eigenschaften nicht erfasst sind.

Profilvorlage löschen

Da Profilvorlagen nicht ewig existieren, muss der Administrator in der Lage sein, sich dieser zu entledigen. Dies erfolgt mittels einer Löschfunktion. Das erfolgt dann sehr häufig, wenn ein Profil aus mehreren Kategorien besteht. Analog besteht also eine Profilvorlage wiederum selbst aus Profilvorlagen. Wenn nun eine der Kategorien nicht mehr durch das Angebot des Ferienclubs angesprochen wird, ist die Haltung entsprechender Daten nicht mehr notwendig. Ergo kann das der Kategorie entsprechende Profil gelöscht werden, wenn keinerlei Referenz auf Unterkategorien beziehungsweise Profildaten mehr besteht.

Profildaten einsehen

Um eine Übersicht über die tendenziellen „Neigungen“ der sich momentan im Ferienclub befindlichen Gäste zu verschaffen, ist der Administrator in der Lage, sich eine Übersicht über alle ad hoc verfügbaren Profildaten zu verschaffen. Inwiefern Anonymität gewährleistet werden muss, soll in dieser Arbeit nicht weiter geklärt werden und hängt von den aktuellen Datenschutzbestimmungen des jeweiligen Einsatzortes ab. Der Nutzen dieser Funktion liegt darin, dass man das Angebot, welches eventuell simultan modifiziert werden muss, optimal dem möglichen Auditorium anpassen kann.

Veranstaltung anlegen

Das oben genannte Angebot besteht aus mehreren Veranstaltungen. Dabei ist der Begriff der Veranstaltung sehr abstrakt gehalten und umfasst alle erdenklichen Formen der gemeinschaftlichen Aktivität. Damit überhaupt ein Angebot entstehen kann, muss der Administrator in der Lage sein, Angebote den Gästen zur Verfügung zu stellen ergo solche anzulegen.

Veranstaltung bearbeiten

Es kann vorkommen, dass Veranstaltungen sich beispielsweise in der Terminierung oder der Lokalität ändern können. Damit diese Änderungen den Gästen ad hoc zugänglich sind, muss der Administrator in der Lage sein, diese Veranstaltungen beziehungsweise deren Eigenschaften im System zu verändern. Eine andere Möglichkeit der Änderung wäre, dass sich der inhaltliche Schwerpunkt einer Veranstaltung gravierend ändert und sich die Kategorienzugehörigkeit verändert.

Veranstaltung löschen

Wenn allerdings eine Veranstaltung überhaupt nicht mehr stattfindet, sollte diese Veranstaltung ad hoc aus dem System entfernt und die Gäste simultan darüber informiert werden. Dies erfolgt über das Löschen einer Veranstaltung, was auch nur dem Administrator vorbehalten ist.

2.5.2.2 Anwendungsfälle des Gastes

Die Aktionen des Gastes sind in dem Beispielszenario relativ beschränkt. Ob weitere Aktionen möglich sind, soll hier nicht ausgeschlossen werden. Jedoch sind die hier genannten signifikant für das gegebene Szenario und somit ausreichend für eine weiterführende Betrachtung.

Profildaten eingeben

Die Daten, mit denen die im Beispielszenario genannte Applikation agiert, werden mit einem vorgegebenen Wert an den Benutzer - den Gast - übergeben. Damit dieser vom Nutzen der Applikation maximal profitieren kann, ist dieser in der Lage, das vorgegebene Profil „seinem“ Profil anzupassen.

Profildaten bearbeiten

Kommt es vor, dass ein Benutzer sich in seinen Neigungen beziehungsweise in seinem Kenntnisstand verändert hat, muss er in der Lage sein, sein Profil anzupassen. Diese Anpassung erfolgt mittels einer Editierfunktion.

Profildaten einsehen

Die Daten, die der Benutzer unter seinem Profil eingegeben hat, können durch diesen auch eingesehen werden.

Einladung lesen

Stellt der Administrator ein neues Veranstaltungsangebot ins System des Ferienclubs, bekommen die Gäste des Clubs eine entsprechende Information. Diese kann durch den Gast eingesehen, jedoch nicht editiert oder gelöscht werden.

Teilnahme annehmen bzw. ablehnen

Wenn der Gast sich dazu entscheidet, eine Einladung, die er bekommen hat, anzunehmen, kann er die Teilnahme an der Veranstaltung durch die Applikation bestätigen oder ablehnen. Ablehnung beziehungsweise Annahme der Einladung zur Teilnahme durch den Benutzer führt zum Löschen der Einladung aus dem System des Benutzers. Im Falle der Annahme erhält der Benutzer jedoch eine entsprechend geartete „Reservierung“.

Reservierung einsehen

Die oben genannte Reservierung kann sich im Laufe der Zeit[2] ändern. Der Benutzer hat selbst keine Einflussmöglichkeit auf die Änderung, kann die aktuellen Informationen jedoch immer einsehen.

Unternehmung einstellen

Doch nicht nur der Administrator ist in der Lage, Veranstaltungen anzubieten. Ein Gast kann zum Beispiel einen Partner zum Tennisspielen suchen. Dazu kann er eine Unternehmung anbieten, die in dem System bereit gestellt wird. Jedoch hat hier auch der Administrator die Rechte der Editierung und des Löschens. Eine Unternehmung ist im Grunde eine Veranstaltung, jedoch sind die Eigenschaften einer Unternehmung nur eine Untermenge der Eigenschaften einer Veranstaltung.

Unternehmung bearbeiten

Analog der Möglichkeit der Bearbeitung einer Veranstaltung durch den Administrator kann auch der Gast die von ihm eingestellte Unternehmung bearbeiten. Auch hier kann es vorkommen, dass Veranstaltungen sich in der Terminierung oder der Lokalität ändern können.

Unternehmung löschen

Wenn ein Gast sich jedoch aus irgend einem Grund - sei es Krankheit, Motivation oder was auch immer - dazu entscheidend, seine Unternehmung zu stornieren, wird diese aus dem System gelöscht, wenn die Menge der involvierten Personen nicht größer als 2 Personen ist. Wenn mehr Personen involviert sind, dann wird allen Personen, die bisher dieser Unternehmung zugesagt haben, die Frage gestellt, ob sie weiterhin dieser Unternehmung nachgehen wollen. Jeder Person, die zustimmt, werden nun die entsprechenden Rechte (Bearbeiten, Löschen) übertragen.

[2] Bis zu einem bestimmten Zeitpunkt, der durch den Administrator in den Eigenschaften einer Veranstaltung als Deadline definiert werden kann, bis zu dem ein Benutzer seine Teilnahme an einer Veranstaltung annehmen bzw. zurückziehen kann.

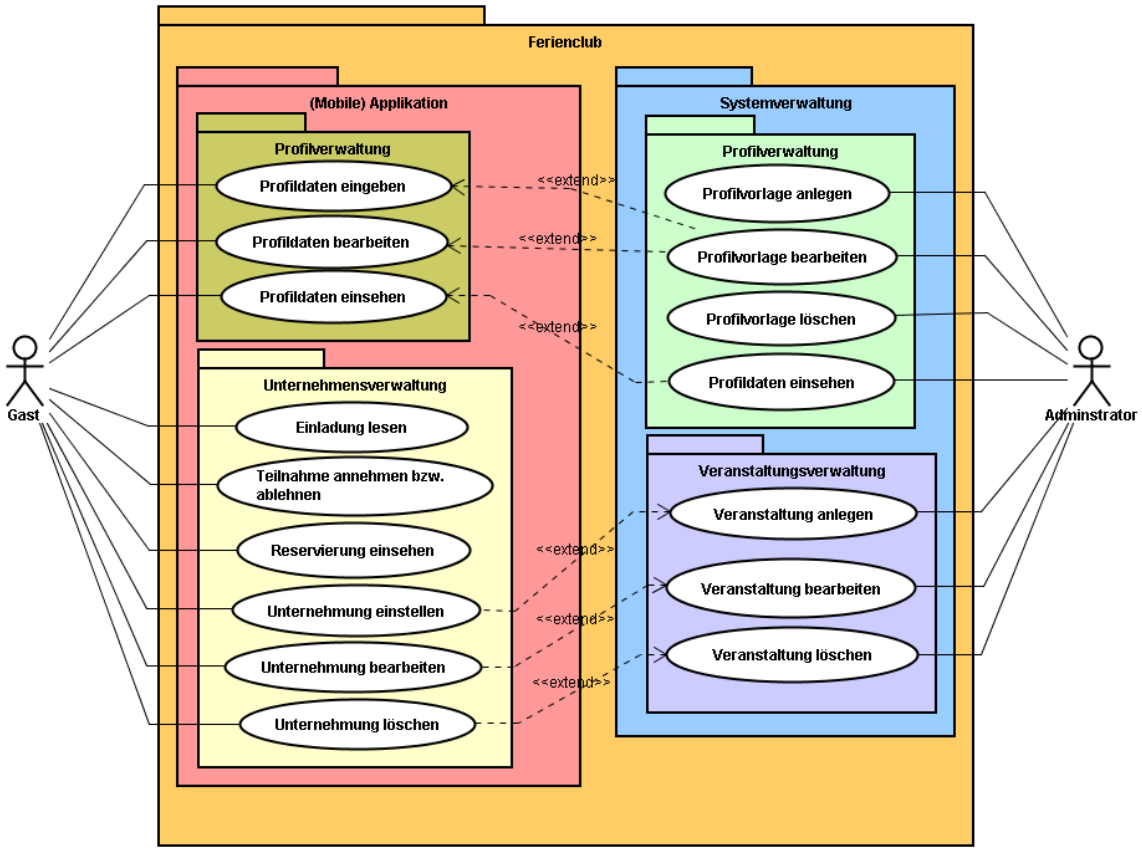

Abbildung 2.1: Anwendungsfälle

2.6 Profile

In der Problemstellung (Kapitel 1.2) wurde der Begriff des Agenten erwähnt. Es handelt sich im herkömmlichen Sinne um eine Person, die in dessen Auftrag und in dessen Sinn handelt. Nun kann ein Agent ausgehend von seiner Intuition versuchen, möglichst im Sinne seines Auftraggebers zu handeln. Jedoch ist der Erfolg dieser Handlungsgrundlage bedenklich, da nicht sichergestellt ist, wann eine Aktion sinngemäß war[3]. Eine Grundlage, welche die Erfolgschance eines Agenten erhöhen, ist das Profil des Auftraggebers (Babic, 2003; Butrynowski, 2005).

2.6.1 Allgemeine Betrachtung

Unter dem Profil einer Person versteht man zum einen eine stark ausgeprägte (charakterliche) Eigenart. Allgemein jedoch werden unter einem Profil diverse Eigenschaften zusammengefasst, die typisch für jemanden oder etwas sind (Drosdowski u. a., 1997). In Bezug auf die zu entwickelnde Applikation ist das Profil des Benutzers maßgeblich für den Erfolg der Applikation.

Das Profil wird dargestellt durch eine Menge von Eigenschaften, die der Benutzer hat. Da jedoch in einem Ferienclub n Gäste zugegen sind, gibt es demnach auch n Profile, da die Äquivalenz von Individuen ausgeschlossen werden kann. Warum das Profil maßgeblich für den Erfolg der Applikation ist, soll im folgenden geklärt werden.

Anwendung

In dem Beispieleszenario (Kapitel 1.4) geht es darum, dass ein Gast zusammen mit anderen Gästen an einem Essen teilnimmt. Die Sitzordnung kann nun chaotisch oder - und das ist eine der Aufgaben der zu entwickelnden Applikation - nach bestimmten Kriterien erfolgen. Hintergrund der systematischen Erstellung einer Sitzung ist die Tatsache, dass der Ferienclub von der Benutzung der Applikation profitieren will. Das ist dann gegeben, wenn der Gast ebenfalls von der Applikation profitiert. Nun gilt es herauszufinden, wann ein Gast von der Benutzung der Applikation profitiert.

[3] Genau genommen gibt es keine Grundlage, die eine Handlung einer Person im Sinne einer anderen zu 100% ausführen lässt. Der Erfolg beruht vielmehr auf Intuition, Erfahrung und suksezzive Approximation der Verhaltensweisen.

Def.: Ein Gast profitiert dann von der Benutzung der Applikation, wenn die Teilnahme an einer Unternehmung mit anderen Gästen zur größtmöglichen Zufriedenheit des Gastes führt. In Bezug auf das Beispielszenario sei größtmögliche Zufriedenheit dann gegeben, wenn sich das Verhältnis zwischen den Gästen in größter Harmonie äußert.

Doch was bedeutet Harmonie in diesem Zusammenhang? Wann besteht zwischen mehreren Personen größtmögliche Harmonie? Grundlage dieses Kriteriums sind die Profile der Benutzer. Um festzustellen, ob mehrere Personen harmonieren, kann man ihre Profile auf Ähnlichkeit hin überprüfen. Dazu ist es jedoch erforderlich, dass die Profile miteinander vergleichbar sind. Das kann dadurch erreicht werden, dass man allen Gäste die gleichen Fragen stellt und deren Aussagen bei Bedarf analysiert und vergleicht. Doch stellt sich an diesem Punkt eine wichtige Frage: In welcher Beziehung stehen Harmonie und Profilähnlichkeit zueinander?

In diesem Punkt gibt es zwei konträre Standpunkte[4]:

1. „Gegensätze ziehen sich an“ vs.
2. „Gleich und Gleich gesellt sich gern“

Im ersten Fall ist es so, dass die Ähnlichkeit minimal sein muss, damit sich unterschiedliche Individuen verstehen. Dabei ist die Kernaussage des zweiten Falls, dass die Ähnlichkeit maximal sein muss. Für diese Arbeit wird die zweite Aussage zu Grunde gelegt. Man könnte sich auch auf die erste Aussage beschränken, was nicht all zu große Auswirkungen auf die Realisierung hätte. Jedoch würde die Berücksichtigung beider Standpunkte den Rahmen der Arbeit sprengen und weit über den Horizont der Informatik hinausgehen, von daher existiert eine bewusste Einschränkung der Untersuchung. Es gilt insbesondere hinsichtlich eines Spezialfalls. Es gibt aus psychologischer Sicht spannende Phänomena hinsichtlich der Harmonie von Individuen gleichen Profils. Zum einen kann es sein, dass die These maximaler Harmonie zutrifft, es kann jedoch auch das genaue Gegenteil eintreffen. Und die Phänomena sind dabei noch nicht einmal nur vom Profil abhängig, da es sein kann, dass eine Person sich mit einer gleichen Profils gut versteht, eine andere jedoch verabscheut, obwohl diese auch das gleiche Profil hat.

Die Wahl des Standpunktes erfolgte jedoch nicht zufällig. Die These, dass sich ähnelnde Individuen gruppieren, ist dadurch belegt, dass die These der sich anziehenden Gegensätze durch die Wissenschaft häufig widerlegt wurde. Wenn also die erste These falsch ist, muss die zweite These stimmen [5]. So lieferte beispielsweise eine Studie der schottischen Universität St. Andrews ein erstaunliches Ergebnis. Frauen sollten darin aus einer Menge von

[4] Diese Darstellung ist sehr vereinfacht und alle, die sich näher mit der Materie beschäftigt haben, mögen mir diese starke Abstraktion verzeihen.

[5] Was sich leicht durch Euklidische Beweisführung bestätigen lässt.

Männergesichtern jenes bestimmen, welche für sie am attraktivsten ist. Dabei bestimmten sie jeweils das Bild, was sie selber zeigte, da die Forscher zuvor die Bilder der Testkandidatinnen in das jeweils männliche Pendant überführt und unter die Auswahl gemischt haben (PM, 2004).

Doch die These gilt nicht nur für Äußerlichkeiten. Laut einer Umfrage der Chicagoer Universität gilt diese These auch bei der Wahl des Sexualpartner, wobei die erfragten Eigenschaften nicht äußerlicher Natur waren. Grundlage ist die Tatsache, dass eine partnerschaftliche Beziehung nicht nur aus Sex besteht. So wirke die Ähnlichkeit anderer Eigenschaften der involvierten Individuen zur äußerlichen Attraktivität zusätzlich anziehend. Je ähnlicher Interessen und Intellekt, desto einfacher das Zusammenleben, so der Tenor der Umfrage. Je mehr Gemeinsamkeiten in Bezug auf

- Bildungsniveau,
- Alter,
- Sozialer und
- Religiöser Vergangenheit

vorhanden ist, desto unproblematischer ist es für die betreffenden Personen, das Leben gemeinsam zu verbringen (Brody, 1994).

Der amerikanische Verhaltensforscher Richard W. Lewak belegte in einer Langzeitstudie, dass neben den oben genannten Übereinstimmungen zusätzlich Ähnlichkeiten bezüglich folgender Attribute das zwischenmenschliche Zusammenleben erleichtern:

- „Rasse“
- Gehaltstufe
- Politische Ansichten
- etc.

Belegt wurde dies ähnlich der Studie der Universität von St. Andrews. Man nahm die getrennte Darstellung von Eheleuten. Testpersonen sollte die zusammengehörenden Partner bestimmen. Dies gelang den Testpersonen nahezu perfekt. Gefühlsmäßig hatten die Probanden eine Rangfolge des „äußeren Eindrucks“ erstellt und demnach Frauen- und Männergesichter einander zugewiesen (Scheppach, 1988).

2.6.2 Mathematische Betrachtung

Es gilt also, Gäste hinsichtlich ihrer Profile zu vergleichen, Ähnlichkeiten zu erkennen, diese Ähnlichkeiten zu bewerten und entsprechend der Bewertung eine Sitzordnung zu erstellen. Diesbezüglich wurde im Beispielszenario erwähnt, dass ein Cluster die Menge an einem Tisch speisenden Personen ist. Die Größe kann dabei zum einen die Anzahl der Sitzgelegenheiten pro Tisch vorgegeben sein. Da im vorangegangenem Abschnitt erwähnt wurde, dass die Benutzer bezüglich ihrer Profilähnlichkeit hin überprüft, verglichen und gruppiert werden, ist zum anderen auch eine selbstorganisierende Clustergröße denkbar.

In dem Fall der selbstorganisierenden Clustergröße werden Grenzen vorgegeben, wie weit Profile hinsichtlich ihrer Ähnlichkeit auseinander liegen dürfen. Daraus folgt, dass sich Gruppen bilden, welche jeweils aus den Profilen bestehen, deren Ähnlichkeitskoeffizient kleiner/gleich der gegebenen Grenze ist. Nachteil der Selbstorganisation ist jedoch, dass es zu „Einsiedlern“ kommen kann. Um dies zu umgehen, kann man einen Mittelweg der beiden Möglichkeiten wählen, indem man eine minimale Clustergröße vorgibt, welche nicht unterschritten werden darf.

Doch wie können Profile hinsichtlich ihrer Ähnlichkeit hin untersucht werden? Dazu bedient man sich folgender Methodik: Anstatt Profile natürlichsprachlich darzustellen, werden die im letzten Kapitel erwähnten Fragen, die allen Gästen gestellt werden, so dargestellt, dass die Antworten eine Selbsteinschätzung der eigenen Neigung darstellt, welche auf den Bereich der natürlichen ganzen Zahlen abgebildet werden kann. So gibt man mit der Frage einen Bereiche möglicher Antworten vor. So definiert sich ein Profil wie folgt:

Def.: Aus mathematischer Sicht versteht man ein Profil als eine Menge Ω natürlicher ganzer Zahlen, deren Mächtigkeit $|\Omega|$ gleich der Menge der gestellten Fragen ist.

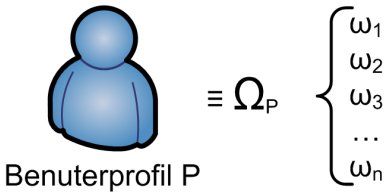

Abbildung 2.2: Gast und Profil

Ergo existieren bei n Gästen in dem Ferienclub n Profile der Größe $|\Omega|$. Die Form der Repräsentanz als $|\Omega|$-dimensionaler (Orts)Vektor $\vec{\Omega}$ wird bei der Realisierung der Applikation verwendet.

Def.: Zur Menge Ω eines Profils P gehört ein (Orts)Vektor $\vec{\Omega}$. Dabei gilt

$$|\Omega| \equiv \text{ Menge der Koordinaten des Vektors } \vec{\Omega}$$

$\vec{\Omega}$ wird wie folgt dargestellt:

$$\vec{\Omega} = \begin{pmatrix} \omega_1 \\ \omega_2 \\ \omega_3 \\ \vdots \\ \omega_n \end{pmatrix}$$

Ähnlichkeit von Profilen

Da die Form der Darstellung von Profilen geklärt ist, muss nun der Forderung der Untersuchung der Ähnlichkeit ρ von Benutzerprofilen nachgegangen werden. Nun gibt es in der Mathematik eine einfache Möglichkeit, die als Vektoren dargestellten Benutzerprofilen auf „Ähnlichkeit“ hin zu untersuchen: die Euklidische Distanz (Butrynowski, 2005).

Dabei werden Profile als Punkte in einem n-dimensionalen Koordinatensystem gesehen, deren Koordinaten durch die Darstellung als Vektor gegeben sind. Die Distanz zweier Vektoren wird als „Euklidischer Abstand“ bezeichnet.

Der Euklidische Abstand ist die Mathematische Definition des normalen Abstands, welcher als Distanzfunktion über zwei Vektoren definiert ist. Der Euklidische Abstand wird als Euklidische Norm des Differenzvektors zwischen den beiden Vektoren berechnet.

$$d(x,y) = |x-y| = \sqrt{(x_1-y_1)^2 + \cdots + (x_n-y_n)^2} = \sqrt{\sum_{i=1}^{n}(x_i-y_i)^2}$$

Es muss beachtet werden, dass Ähnlichkeit und Distanz sich antiproportional zueinander verhalten.

$$\rho = \frac{1}{d(x,y)}$$

Die Euklidische Norm (auch 2-Norm genannt) ist die natürliche (Euklidische) Länge eines Vektors $\vec{x}$:

$$\|\vec{x}\|_2 = \sqrt{|x_1|^2 + \cdots + |x_n|^2}$$

Sie ist ein reiner Zahlenwert (Skalar). Dieser Skalar entspricht einer ganz gewöhnlichen Länge. Die kürzeste Verbindung zwischen zwei Punkten in der euklidischen Norm ist also eine Strecke, die durch sie induzierte Topologie ist die des Euklidischen Raumes.

Da der Euklidische Abstand eine Norm ist, gilt stets,

- dass der Abstand eines Punktes zu sich selbst Null ist,
- er richtungsunabhängig ist und
- die Dreiecksungleichung.

Neben dem Euklidischen Abstand gibt es eine Reihe weiterer Abstandsmaße, die jedoch in dieser Arbeit nicht weiter betrachtet werden (netlexikon 2005).

Vollständigkeit

Wenn man sich an dieser Stelle das Beispielszenario in Erinnerung ruft (Kapitel 1.4), wird folgende Situation klar: Es wird die Einladung zum einem gemeinsamen Mahl an alle Gäste des Ferienclubs versandt. Immer dann, wenn ein Gast die Teilnahme bestätigt, erfolgt eine Berechnung der Ähnlichkeit ρ seines Profils in Bezug auf die Profile aller vor ihm angemeldeten Gäste und damit einhergehend die Bestimmung, wie weit die einzelnen Profile ω_i auseinander liegen. Geht man davon aus, dass diese Informationen, wenn sie einmal berechnet worden sind, nicht erneut berechnet werden müssen, gilt folgendes:

Def.: Es existiert ein in jedem Moment der Bestätigung einer Einladung zu einem gemeinsamen Mahl ein vollständiger, gewichteter Graph K_n, wobei gilt,

1. dass n die Anzahl der Gäste, die eine Einladung angenommen haben ist,
2. $n \subseteq N$, wobei N die Anzahl der Gäste in dem Ferienclub ist,
3. jeder Ecke $v_i \in V$ ein Profil eines Gastes $\Omega_i \in G$ zugeordnet ist und
4. die Gewichtung der Kante $e_i = \{v_i, v_j\} \in E$ äquivalent der Euklidischen Distanz $d(\Omega_i, \Omega_j)$ der inzidenten Ecken v_i und v_j, dargestellt durch die den Ecken v_i und v_j zugeordneten Vektoren $\vec{\Omega_i}$ und $\vec{\Omega_j}$ der Profile Ω_i und Ω_j, ist.

Dieser Graph wird im folgenden auch als Distanzgraph bezeichnet.

In dem folgenden Beispiel haben fünf Gäste g_i diese Einladung angenommen. Man erkennt, dass zwischen jedem Gast eine Verbindung besteht.

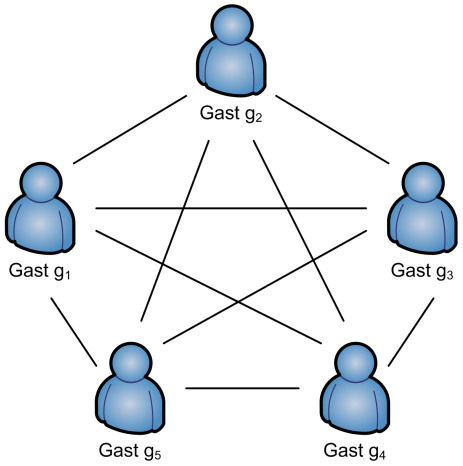

Abbildung 2.3: Das Gästeverhältnis

Berücksichtigt man bei der Darstellung alle oben genannten Fakten, leidet die Übersicht mit zunehmender Teilnehmerzahl erheblich. Die folgende Darstellung ist bezüglich des Informationsgehaltes zwar abstrakt, dennoch vollständig. In Zukunft wird jedoch aus Rücksicht auf die Übersichtlichkeit nur noch die Gewichtung der Kanten und der Profilname, jedoch nicht die Profilwerte in dem Graphen K_n aufgeführt.

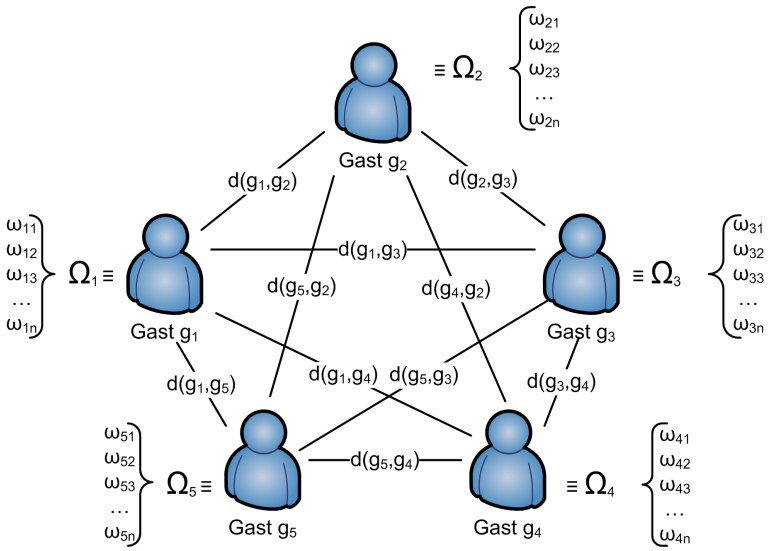

Abbildung 2.4: Alle Informationen des Gästeverhältnisses

Beispiel In Bezug auf das Beispielszenario seien fünf Gäste im Ferienclub: Finja, Paul, Tom, Louis und Lilly. Diese Gäste habe ihr Profil vervollständigt. Es wurde die Neigung zum

Fußball und zum Schwimmen angegeben, wobei die Ausprägung der Eigenschaften einen ganzzahligen Wert zwischen 1 (Totale Abneigung) und 10 (Totale Zuneigung) annehmen kann.

Gast	Ausprägung	
	Schwimmen	Fußball
Finja	9	10
Lilly	2	3
Paul	6	5
Tom	3	7
Louis	1	9

Tabelle 2.1: Die Gäste und ihre Profildaten

Beispielhaft wird die Ähnlichkeit der zwei Gäste Finja und Lilly berechnet. Die Berechnung der anderen Distanzen erfolgt analog zur Beispielrechnung.

$$\begin{aligned} d(\Omega_F, \Omega_{Li}) &= |\vec{\Omega_F} - \vec{\Omega_{Li}}| \\ &= \sqrt{(\omega_{F1} - \omega_{Li1})^2 + (\omega_{F2} - \omega_{Li2})^2} \\ &= \sqrt{(9-2)^2 + (10-3)^2} \\ &= \sqrt{7^2 + 7^2} \\ &\approx 9,90 \end{aligned}$$

Für die Gäste ergibt sich nach der Berechnung der Graph K_5 aus Abbildung 2.5.

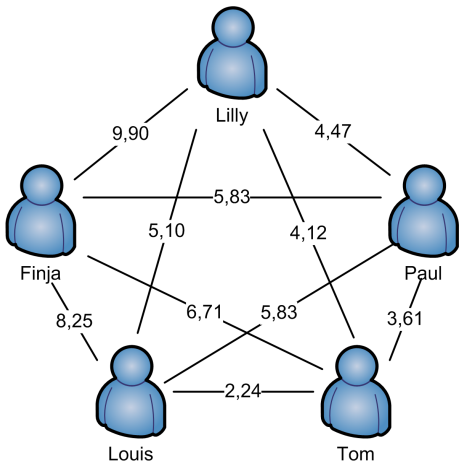

Abbildung 2.5: Der Distanzgraph der Gäste

Manipulation

Bei der Berechnung des Euklidischen Abstands, welcher die Kantengewichtung in dem vollständigen gewichteten Graphen K_n darstellt, werden alle Elemente ω_i und ω_j der zu Grunde liegenden Profile Ω_i und Ω_j beziehungsweise die Koordinaten ω_i und ω_j der inzidenten Vektoren $\vec{\Omega_i}$ und $\vec{\Omega_j}$ gleichberechtigt involviert. Daraus lässt sich folgendes Faktum ableiten:

$$\vec{\Omega}_{i \in n} = \begin{pmatrix} \lambda \bullet \omega_1 \\ \lambda \bullet \omega_2 \\ \lambda \bullet \omega_3 \\ \vdots \\ \lambda \bullet \omega_{|\Omega|} \end{pmatrix}$$

λ ist dabei eine Konstante, die für alle Elemente $\omega \in \Omega$ gilt. Auf Basis dieser Überlegungen und der oben genannten Art der Bestimmung der Tischordnung lässt sich diese leicht dahingehend manipulieren, dass man den in diesem Fall kommunikativen Schwerpunkt derselbigen von außen bestimmt. Dafür ist es notwendig, den Konstanten Faktor λ nur für jene Elemente $\omega \in \Omega$ aufrecht zu erhalten, die einen besonderen Stellenwert bei der Zusammenstellung einer Tischordnung genießen sollen. Alle anderen können mit einem beliebigen Wert $\chi < 1$ multipliziert werden. Dieses Verfahren könnte man beispielsweise anwenden, wenn man lauter Programmierer hat, bei denen man möchte, dass die Eigenschaft der Erstellung grafischer Benutzeroberflächen zu 100% berücksichtigt werden soll.

Beispiel Es soll das schon bekannte Szenario mit den wohlbekannten fünf Gästen gelten. Jedoch will man den Fokus mehr auf Fußball ($\lambda_F = 1$) legen und Schwimmen $\lambda_Sch = 0,2$ weitestgehend außer Acht lassen. Zur näheren Betrachtung werden die Profile der Gäste Finja und Louis herangezogen.

$$\begin{aligned} d(\Omega_F, \Omega_{Lo}) &= |\vec{\Omega_F} - \vec{\Omega_{Lo}}| \\ &= \sqrt{((\lambda_{Sch} \cdot (\omega_{F1} - \omega_{Lo1}))^2 + (\lambda_F \cdot (\omega_{F2} - \omega_{Lo2}))^2} \\ &= \sqrt{(0,2 \cdot (9-1))^2 + (1 \cdot (10-8))^2} \\ &= \sqrt{2,56^2 + 2^2} \\ &\approx 2,56 \end{aligned}$$

Bei Berücksichtigung aller Gäste ergibt sich nach der Berechnung folgender Graph K_5

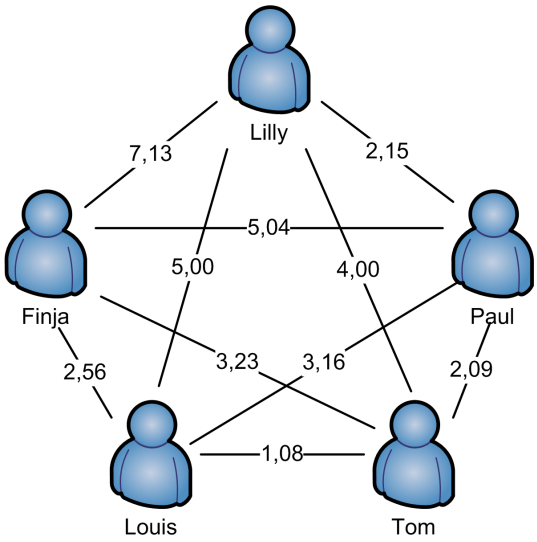

Abbildung 2.6: Der Distanzgraph der Gäste bei Manipulation

Man kann deutlich erkennen, dass sich zwischen den Personen, die sich vor der Manipulation in Bezug auf Fußball nicht „grün“ waren, nicht viel geändert hat. Bei jenen Gästen, bei denen die Meinung nur in Bezug auf Schwimmen auseinander gegangen ist, hat sich das Verhältnis deutlich gebessert.

Eine andere Möglichkeit ist, dass man als Faktor die Hörweiten der adjazenten[6] Individuen eines Clusters mit einbezieht. Das bedeutet, dass man einen Cluster, der durch Gleichberechtigung der Elemente $\omega \in \Omega$ nicht in der Menge möglicher Resultate gelangt wäre, nun in Betracht ziehen kann. Umgangssprachlich formuliert heißt es, das Personen, die offensichtlich nicht miteinander klar kämen, nun doch an einem Tisch speisen. Dabei werden die Brennpunkte jedoch so plaziert, dass sie sich nicht in Hörweite zueinander befinden.

Partielles Matching

Ausgehend von den Überlegungen der vorangegangenen Kapitel ist man nun in der Lage, Überlegungen dahingehend anzustellen, wie man den zu Grunde liegenden Graphen in eine Tischordnung überführen kann. Wichtig ist hierbei die Berücksichtigung der Clustergröße. Es muss nun ein Algorithmus gefunden werden, der aus einem gegebenen vollständigen gewichteten Graphen K_n Komponenten extrahiert, die folgenden Bedingungen genügen:

[6] Adjazent beschreibt in der Graphetheorie das Verhältnis zweier durch eine oder mehrere Kanten verbundene Ecken zueinander.

1. Die Summe aller Kantengewichten ist minimal,
2. die Anzahl der Elemente einer Komponente überschreitet die vorgegebene Clustergröße nicht bzw.
3. die Differenz der Kantengewichte unterschreitet eine gewisse Grenze nicht (im Fall selbstorganisierender Cluster)

Problematik

Offensichtlich hat man es hier mit einem Problem zu tun, für das es mehr als eine Lösung gibt, vorausgesetzt, die Menge der involvierten Personen ist größer 1 und übersteigt die Anzahl der Sitzgelegenheiten pro Tisch. Doch welche der Lösung soll man nehmen? Es gibt zwei unterschiedliche Sichtweisen, an dieses Problem heranzugehen. Zum einen kann man nur die optimale Lösung akzeptieren. In diesem Fall wird die zu Grunde liegende Problemstellung auch als Suchproblem bezeichnet. Dies setzt jedoch voraus, dass es eine optimale Lösung gibt.

Zum anderen kann man eine Lösung auch ein Qualitätsmerkmal - die Güte - zuweisen. Dabei handelt es sich bei der Güte meist um eine reelle Zahl. Bei dieser Art der Problembetrachtung geht es weniger darum, eine einzige optimale Lösung zu finden und als Lösung zu akzeptieren. Vielmehr wird bei diesem Ansatz auch schon eine zufriedenstellende Lösung aktzeptiert. Dabei muss natürlich festgelegt werden, was zufriedenstellend eigentlich heißt. Außerdem ist es im Vorfeld zu klären, ob es sich um ein Minimierungs- oder ein Maximierungsproblem handelt. Diese Form der Problematik bezeichnet man als Optimierungsproblem.

Ausgehend davon, dass man nicht weiß, ob es überhaupt eine optimale Lösung gibt[7], kann das Problem in die Gruppe der Optimierungsprobleme eingeordnet werden. Nun kann das Problem der Gruppenkonfiguration aus zwei Blickwinkeln[8] betrachtet werden, und zwar als

- Reihenfolgeproblem oder als
- Clusteringproblem.

Reihenfolgeproblem

Bei Reihenfolgeproblemen geht es im Prinzip darum, bestimmte Dinge in einer gewissen Reihenfolge zu tun, wobei keine Sache mehr als einmal getan werden darf. Es kommt aus dem Bereich der Graphentheorie und das bekannteste Beispiel ist das *Problem des Handlungsreisenden (TSP)*. Das Problem des Handlungsreisenden kann folgendermaßen beschrieben werden:

[7] Man betrachte nur den Fall, dass alle involvierten Personen die gleiche Ähnlichkeit haben.

[8] Es ist nicht ausgeschlossen, dass es mehr als zwei Sichtweisen gibt, jedoch sollen in dieser Arbeit nur zwei betrachtet werden, da die unterschiedliche Sichtweisen nicht direkter Gegenstand dieser Arbeit sind.

In einem Ort v_1 wohnt ein Handlungsreisender. Er möchte alle Kunden in den Orten v_2 bis v_n besuchen und am Ende seiner Reise wieder zu Hause sein. Die Entfernung zwischen den Orten ist größer *null* und es gibt von jedem Ort zu jedem Ort eine Straße. Jeder Ort darf jedoch nur einmal besucht werden. Die Frage lautet nun, in welcher Reihenfolge die Orte besucht werden müssen, damit der vom Handlungsreisenden besuchte Weg minimal wird.

Es handelt sich demnach um einen vollständigen Graphen K_n, bei dem die Ecken die Städte und die Kanten die Straßen zwischen den Städten darstellen. Die zu bewältigende Aufgabe ist es, einen sogenannten *Hamiltonkreis* zu finden, bei dem die Kantengewichtung minimal ist. Ein Hamiltonkreis - auch als Rundreise bezeichnet - ist ein Kreis, der jede Ecke eines Graphen genau einmal enthält. Das Problem an Hamiltonkreisen ist, dass es für die Konstruktion eines Hamiltonkreises keinen polynomialen Algorithmus gibt (Maas und Klauck, 1999).

Man kann das Problem des Handlungsreisenden einfach auf das Beispielszenario übertragen. Die Städte entsprechen den Gästen und die Straßen sind äquivalent der Distanz zwischen den inzidenten Gästen. Wie erwähnt, gibt es keinen bekannten polynomialen Algorithmus zur Lösung, aber es gibt zahlreiche Ansätze.

Hopfield Netze

Einer dieser Ansätze sind die sogenannten Hopfield-Netze. Es handelt sich dabei um sogenannte autoassoziative Netze. Zwar sind diese im Bereich der neuronalen Netze angesiedelt, entsprechen aber nicht wirklich der Definition neuronaler Netze, da alle Neuronen untereinander verbunden sind und keine Ein- beziehungsweise Ausgabeschicht existiert.

Eine Quasi-Eingabe wird durch initiale Aktivierung der Neuronen erreicht. Das hat zur Folge, dass das Netz so lange arbeitet, bis keine Änderung zwischen den Zuständen s_{n-1} und s_n mehr auftritt. Der Aktivierungszustand s_n stellt dann die Quasi-Ausgabe des Netzes dar.

Kernpunkt von Hopfield-Netzen ist die *Energiefunktion E*. Diese verringert sich von Arbeitsschritt zu Arbeitsschritt. Sobald sich das Netz im Zustand s_n befindet, muss E ein Minimum erreicht haben. Jetzt kann die Energiefunktion auf das jeweils zu lösende Problem angepasst werden.

Beispiel Man möchte untersuchen, in welcher Reihenfolge man die Städte Hamburg, Berlin, München, Köln und Bremen besuchen muss, um einen Hamiltonweg[9] mit minimaler Kantengewichtung zu erhalten. Man kann dies mittels eines Hopfieldnetzes modellieren. Dabei

[9] Der Unterschied zwischen einem Hamiltonkreis und einem Hamiltonweg ist, dass bei dem Weg Anfangs- und Endpunkt von einander verschieden sind. Bei dem Kreis sind sie identisch.

lassen die sich oben genannten Städte in der gleichen Reihenfolge durch ein Netz mit folgender Aktivierung darstellen[10]:

Stadt	Reihenfolge				
	1	**2**	**3**	**4**	**5**
Hamburg	1	0	0	0	0
Bremen	0	0	0	0	1
Köln	0	0	0	1	0
München	0	0	1	0	0
Berlin	0	1	0	0	0

Tabelle 2.2: Hamiltonweg mittels Hopfield-Netz

Die Energiefunktion E soll mit Hilfe diese Netzes minimiert werden. Es gilt, dass

- E nur für die Lösungen minimal sein darf, die genau eine 1 in jeder Zeile und Spalte haben und
- E muss für Lösungen mit kürzerer Pfadlänge geringer sein als für solche mit längerer Pfadlänge.

Es kann jetzt die Funktion zur Erfüllung dieser Bedingungen definiert werden. Da an dieser Stelle Hopfield-Netze nur grob vorgestellt werden und für den restlichen Teil der Arbeit nicht von Belang ist, wird diese Funktion nur verbal erklärt und nicht näher erläutert.

Die oben genannte Funktion besteht aus vier Termen:

- Term 1 darf nur dann 0 ergeben, wenn jede Zeile eine 1 enthält. Somit ist sichergestellt, dass jede Stadt genau einmal besucht wird.
- Term 2 darf nur dann 0 ergeben, wenn je Spalte genau eine 1 vorkommt. Damit stellt man sicher, dass man nicht in mehreren Städten gleichzeitig ist.
- Term 3 ergibt nur dann 0, wenn die Anzahl der Zahl 1 n-mal vorkommt. Dabei steht n für die Anzahl der Städte. Es wird sichergestellt, dass auch wirklich jede Stadt besucht wird.
- Term 4 gibt die Länge der aktivierten Strecke an.

Somit ist ersichtlich, dass man Hopfield-Netze auch zur Lösung der Gruppenkonfiguration benutzen kann. Hierzu muss nur bekannt sein, auf wieviele Gruppen die Personen zu verteilen sind (siehe Tabelle 2.3). Ebenfalls muss eine entsprechende Energiefunktion definiert werden.

[10] Eine 1 kennzeichnet jeweils ein aktiviertes Neuron

Person	Gruppe		
	1	2	3
1	1	0	0
2	0	0	1
3	0	0	1
4	0	1	0
5	1	0	0
6	0	1	0
7	0	0	1
8	1	0	0
9	0	1	0

Tabelle 2.3: Gruppenkonfiguration mittels eines Hopfiel-Netzes

Analog zur vorgestellten Energiefunktion E besteht die Energiefunktion $E_G k$ aus folgenden vier Termen:

- Term 1 ergibt nur dann 0, wenn jede Zeile eine 1 enthält. Dies stellt sicher, dass eine Personen nicht in mehreren Gruppen gleichzeitig ist.
- Term 2 ergibt nur dann 0, wenn die Anzahl des Wertes 1 pro Spalte größer gleich 2 ist. Damit wird sichergestellt, dass eine Gruppe aus mindestens zwei Personen besteht.
- Term 3 ergibt nur dann 0, wenn der Wert 1 in der Matrix genau n-mal vorkommt. Dabei steht n für die Anzahl der vorhandenen Profile. Somit wird auch jede Person einer Gruppe zugewiesen.
- Term 4 gibt die Summe der Profildistanzen pro Gruppe an.

Die Komplexität der Implementierung von Hopfield-Netzen und deren Performanz sorgen jedoch dafür, dass man eher nach einer Alternative Ausschau hält. Die in dieser Arbeit betrachtete Alternative ist die Clusteranalyse.

Clusteringproblem

Wie eingangs erwähnt, gibt es neben der Betrachtung als Reihenfolgeproblem noch eine weitere Betrachtungsweise: Gruppenkonfiguration als Clusterproblem. Bei Clusterproblemen geht es darum, aus einer beliebigen Menge von Objekten Gemeinsamkeiten zu bestimmen und entsprechende Klassen zu bilden. Konkret befinden sich diese Objekte in einem Mehrdimensionalen Raum. Dieser Raum wird auch als Merkmalsraum bezeichnet.

Ordnungskriterium ist hier die Ähnlichkeit der Objekte. Man spricht in diesem Zusammenhang auch von Homogenität. Die Objekte, die nicht einer Klasse angehören, werden demnach als heterogen - unähnlich - bezeichnet.

Die Clusteranalyse findet vielerlei Anwendung. Ein beliebtes Anwendungsgebiet ist zum Beispiel die Marktsegmentierung. In diesem Fall werden die Kunden eines Unternehmens näher betrachtet. Man unterteilt diese dann gemäß einiger Eigenschaften in Segmente ein. Beispielsweise könnten es folgende Merkmale sein, die der Segmentierung dienen:

- Vorlieben,
- Alter,
- Geschlecht,
- etc.

Wie gesagt, es sind beliebig viele Kriterien der Segmentierung möglich. Wichtig ist in diesem Fall die Erkenntnis, dass sich unsere Profile also mit relativ einfachen Mitteln gruppieren lassen. Ein Algorithmus, der in Bezug auf die Clusteranalyse eine weite Verbreitung erfahren hat, ist der sogenannte *k-means-Algorihtmus*.

Der k-means-Algorithmus

Die Voraussetzungen zur Anwendung des k-means-Algorithmus sind durch die vorangegangenen Betrachtung quasi schon gegeben. Es werden folgenden Dinge zur Anwendung des Algorithmus benötigt:

Bedingung	Erfüllt?	Anmerkung
1.Es existieren n Objekte.	✓	Die vorhandenen Profile.
2.Es wird eine Anzahl k der zu generierenden Cluster angegeben.	✓	Die Anzahl der zu bildenden Gruppen
3.Es existiert eine Abstandsfunktion.	✓	Euklidische Distanz
4.Es existiert eine Funktion der Berechnung des Mittelpunktes eines Clusters.	✓	Eine einfache Funktion zur Berechnung des Mittelpunktes eines Wertebereichs.

Tabelle 2.4: Voraussetzungen für den k-means-Algorithmus

Der k-means-Algorithmus besteht aus folgenden Schritten:

1. Bestimme zufällig k Punkte. Dies sind nun Clusterzentren.
2. Ordne jedes Objekt dem nächstgelegenen Zentrum zu.

3. Berechne für jedes Cluster das Zentrum neu.
4. Falls sich die Zentren nicht mehr ändern: Abbruch, sonst weiter bei 2.

Dieser Algorithmus bringt uns dem Ziel, Gruppen zu konfigurieren, beträchtlich näher. Die Einfachheit des Algorithmus präferiert ihn als Mittel zum Zweck. Dennoch gibt es ein Problem.

Fuzzy-Logik

Man halte sich den Gegenstand der Betrachtung noch einmal vor Augen. Es geht darum, dass eine Person sich selbst in Bezug auf vorher definierte Neigungen, Fertig- und Fähigkeiten einschätzt. Diese Neigungen werden dann über ein wie auch immer geartetes Frontend durch Eingabe auf den Bereich der natürlichen Zahlen abgebildet und gespeichert. Bei erneutem Rekapitulieren dieser Sätze stößt man auf ein Problem: Die Abbildung einer Neigung auf eine ganzen Zahl bedeutet, diese Neigung exakt zu machen. Dabei handelt es bei der Neigung um eine Selbsteinschätzung, also einen eher vagen denn exakten Wert, der zu dem individuell ist. Die nächste Hürde ist die Konfiguration der Gruppen. Die Zuordnung auf Grund der Ähnlichkeit ist ein zusätzliches vages Faktum. Der Umgang mit vagem Wissen kann jedoch durch Fuzzy-Logik bewältigt werden.

Die Motivation hinter der sogenannten Fuzzy-Logik ist die Tatsache, dass die meisten Menschen ihre Gedanken in irgendeiner Form linguistisch formulieren. Dabei sind diese Gedanken in den seltensten Fälle mathematisch exakt. Trotzdem liefern solche unscharfen Aussagevariablen durchaus brauchbare Schlussfolgerungen, wenn man logische Operatoren auf sie anwendet. Mit all solchen unscharfen Aussagen und unscharfen Schlussfolgerungen beschäftigt sich die Fuzzy-Logik.

Bei der Fuzzy-Logik handelt es sich im Grunde um eine Erweiterung der allgemeinen Mengenlehre und Logik. Dabei geht die Fuzzy-Logik davon aus, dass alles nur zu einem gewissen Grad zutrifft. Aussagen haben also eine gewisse Elastizität bezüglich ihres Wahrheitswerts. Als Sonderfall der Fuzzy-Logik hat die Boolesche Algebra eine Elastizität gleich *null*.

Auch wenn Fuzzy-Systeme mit unscharfen Größen arbeiten, sind die zu Grunde liegenden Regeln exakt und eindeutig beschrieben. Somit ermöglicht die Fuzzy-Logik die Darstellung und Verarbeitung von Unsicherheiten mit mathematischen Mitteln (Friedrich, 1997).

Betrachtet man die klassische Mengenlehre, so ist die Teilmenge T eine Menge M wohldefiniert: Man kann für jedes Element m der Menge M eindeutig sagen, ob es zur Teilmenge T gehört, oder nicht.

$$f(m) := \begin{cases} 1 & \text{für} \quad m \in T \subseteq M \\ 0 & \text{sonst} \end{cases}$$

Man spricht in diesem Fall auch vom **Zweiwertigkeitsprinzip der Zugehörigkeit**. In der mathematische Logik wird man ebenfalls mit dieser Zweiwertigkeit konfrontiert: in der Form von *„wahr“* oder *„falsch“*. Wie bereits erwähnt, gilt die Zweiwertigkeit bei der Fuzzy-Logik nicht. Hier spricht man von einer **Mehrwertigen Logik**. Durch eine ***charakteristische Funktion*** χ kann jedoch beschrieben werden, ob ein Wert x in einem bestimmten Intervall $[\alpha, \beta]$ liegt.

$$\chi(x) := \begin{cases} 1 & \text{für} \quad x \in [\alpha, \beta] \\ 0 & \text{sonst} \end{cases}$$

Es bietet sich also an, eine Menge mit relaxierten Zugehörigkeitswerten durch eine charakteristische Funktion zu beschreiben, die Werte zwischen 0 und 1 annehmen kann. Die charakteristische Funktion χ wird häufig auch als Zugehörigkeitsfunktion μ bezeichnet. Diese beiden Begriffe sind im folgendem synonym.

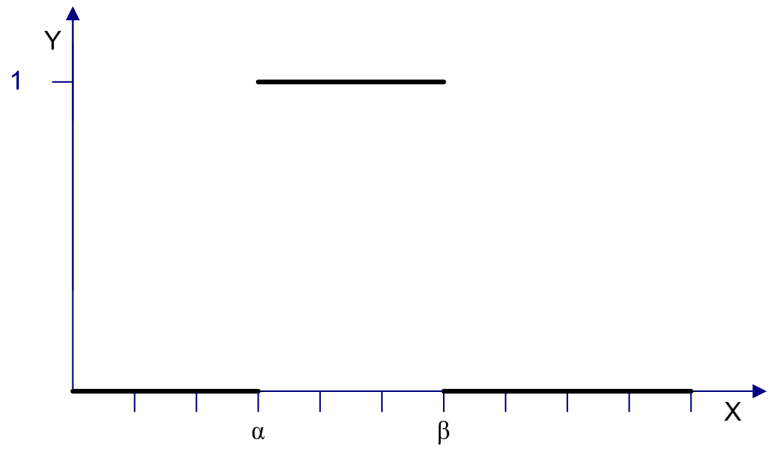

Abbildung 2.7: Charakteristische Funktion χ

Def.: Es gibt eine Grundmenge G. Eine Abbildung

$$f : G \to [0, 1]$$

wird **Fuzzy Menge** in G genannt. Die Abbidlung f wird auch als **Zugehörigkeitsfunktion** der Fuzzy Menge bezeichnet. Den Wert dieser Funktion bezeichnet man als **Zugehörigkeitswert**.

Beispiele

- Es geht um Körpergröße. Es gibt die die Aussage „*Helmut ist 190 cm groß.*“. Dies ist eine exakte Aussage. Dennoch gibt es in Bezug auf diese Aussage eine Unschärfe: Zählt Helmut mit dieser Körpergröße zu den großen Menschen?
- Die Aussage „*Helmut ist zwischen 180 und 190 cm groß.*“ ist vage. Doch sind hier die Intervallgrenzen exakt gegeben.

An dieser Stelle kommen die Fuzzy Mengen ins Spiel. Dabei geht es darum, einer linguistischen Variablen wie zum Beispiel „Körpergröße“ einen Wertebereich - hier Zentimeter - zuzuweisen.

$$\text{Körpergröße} = \{0, 1, 2, \ldots, 250\}$$

Man kann nun für den vagen Begriff „*groß*“ eine Zugehörigkeitsfunktion $\mu_{\text{groß}}$ bestimmen:

$$\mu_{\text{groß}} : \text{Körpergröße} \rightarrow [0, 1]$$

Wenn man nun ein Element $k \in$ Körpergröße daraufhin überprüfen will, ob diese Größe „groß“ ist, übergibt man das Element der Funktion und bekommt

$$\mu_{\text{groß}}(k) = \begin{cases} 1 & \text{falls } \alpha \leq k \leq \beta \\ 0 & \text{sonst} \end{cases}$$

Dabei geben α und β die Grenzen eines Intervalls an, welches vorher definiert wurde und die Grenzen des Bereichs der „großen“ Größen exakt angibt. Diese Form der Funktion entspricht jedoch eher der mathematischen Logik.

Man kann jedoch $\mu_{\text{groß}}$ auch so wählen, dass der Wert der Funktion in dem Intervall $[0, 1]$ liegt. Dabei besagt 0, dass eine Größe überhaupt nicht „groß“ ist. Eine 1 würde bedeuten, dass diese Größe „perfekt groß“ ist. Normalerweise liegt das Ergebnis r der Funktion $\mu_{\text{groß}}$ dazwischen und gibt den Grad an, mit dem das zu untersuchende Element k „groß“ ist.
In der Praxis hat sich hier die Dreiecksform bewährt. Der Funktionswert $\mu_{\text{groß}}$ wird hier durch die Dreiecksform bestimmt (Heinsohn und Socher-Ambrosius, 1999; Frank, 2002).

Die Zugehörigkeitsfunktion sieht in diesem Fall aus wie folgt:

$$\mu_{\text{groß}}(k) = \begin{cases} 1 & \text{für } k = z \\ \frac{k-\alpha}{z-\alpha} & \text{für } k \in [\alpha, z[\quad \alpha < z \\ \frac{-k+\beta}{-z+\beta} & \text{für } k \in]z, \beta] \quad z < \beta \\ 0 & \text{sonst} \end{cases}$$

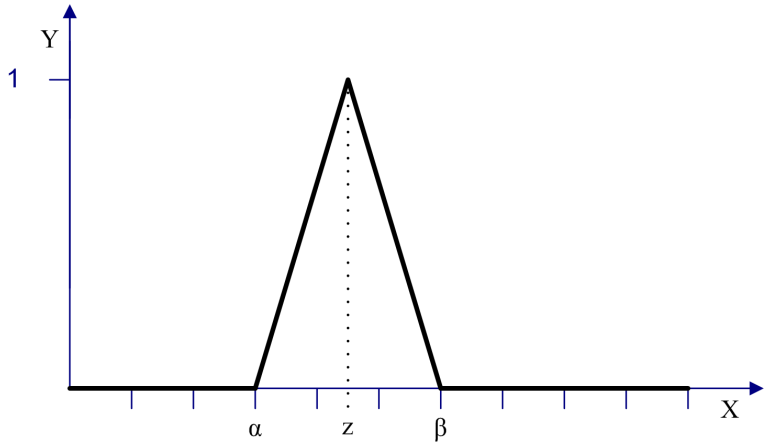

Abbildung 2.8: Dreiecksform

Man hat an dieser Stelle also das Rüstzeug, um die Profile mittels Fuzzy-Logik und Fuzzy-Mengen zu beschreiben. Würden man sich an dieser Stelle der entsprechenden Umsetzung widmen, wäre die Anzahl der Fuzzy-Mengen gleich des Umfangs eines Profil. Der Umfang des Profils entspricht dabei der Menge der gestellten Fragen. Somit steht jeweils eine Fuzzy-Menge in direkter Assoziation mit einem Profilcharakteristikum. Der Wertebereich der assoziierten Fuzzy Menge entspricht dem Wertebereich, auf den die Antwort der assoziierten Frage abgebildet wird.

Fuzzy-Clusteranalyse

Es gibt jedoch noch ein weiteres Problem, und zwar in Bezug auf die Clusteranalyse. Man versucht mittels der Clusteranalyse eine Unterteilung der vorhandenen Objekte in Klassen. Dabei wird jedoch vorausgesetzt, dass jedes Objekt nur Mitglied einer Klasse sein kann. Das trifft in Bezug auf die Profile nicht zu. Jedes charakteristische Merkmal steht im Grunde für eine Klasse, seine individuelle Ausprägung für ihren Wert. Es kann jedoch sein, dass ein Individuum in mehreren Klassen ist. Deshalb wird die Clusteranalyse um Fuzzy-Anteile erweitert. Ein populäres Anwendungsgebiet ist beispielsweise die Mustererkennung. Das wohl populärste und am häufigsten verwendete Verfahren wird im Folgenden kurz vorgestellt.

Fuzzy C-means (FCM)

Der bereits erwähnte k-means-Algorithmus kommt uns an dieser Stelle sehr gelegen, denn man kann ihn mit wenig Aufwand um „Fuzzyness“ erweitern. Hier zu wird die Fuzzy Logic in den bestehenden Algorithmus integriert.

Def.: Die zu minimierende Funktion $J(m)$ ist die Distanz eines beliebigen Datenpunktes zu einem Clusterzentrum gewichtet nach dem Grad der Zugehörigkeit dieses Punktes diesem Clusterzentrum.

$$J(m) = \sum_{k=1}^{K} \sum_{i=1}^{N} u_{i(k)}^{m} e_{i(k)}^{2}$$

wobei gilt

$$u_{i(k)} = \frac{1}{\sum_{l=1}^{K} \left[\frac{e_{i(k)}}{e_{i(l)}} \right]^{\frac{2}{m-1}}}$$

Dabei sind folgende Definitionen gemacht worden:

- $m :=$ Parameter zur Bestimmung der Fuzzyness. Je höher m, desto höher die Fuzzyness („Verteiltheit“). Standardwert ist 2.
- $e_{i(k)} :=$ Distanz eines Punktes zum Clusterzentrum.
- $u_i(k) :=$ „Zugehörigkeitswert“ eines Punktes zu einem Cluster, $u_i(k) \in [0, 1]$.
- $k :=$ Anzahl der Cluster.

Wie der k-means-Algorithmus arbeitet auch der Fuzzy C-means-Algorithmus iterativ:

1. Berechnung der Clusterzentren $z_j = \frac{\sum_{i=1}^{n} u_{ij}^2 x_i}{\sum_{i=1}^{n} u_{ij}^2}$
2. Neuberechnung der Distanzen $d(x_i, z_j) = (x_i - z_j)' (x_i - z_j)$
3. Aktualisierung eine Ähnlichkeitsmatrix U $u_{ij} = \frac{1}{\sum_{r=1}^{k} \frac{d\left(x_i, z_j\right)}{d\left(x_i, z_r\right)}}$
4. Wiederhole Schritt 1 bis 3 solange, bis $\|\Delta U\| < \epsilon$

Man erkennt leicht eine gewisse Ähnlichkeit zwischen den beiden Algorithmen. Die Voraussetzungen für den Fuzzy C-means-Algorithmus sind:

- Ein Set aus Daten,
- die Anzahl der zu bildenden Cluster,
- eine Distanznorm,
- der Fuzzyness-Parameter und

- die Abbruchbedingung[11].

Die Ausgabe der beiden Verfahren ist ebenfalls sehr ähnlich. Der Unterschied hier besteht jedoch darin, dass die Zugehörigkeit der Objekte zu den Clustern fuzzy ist. Das bedeutet, das sowohl die Zugehörigkeitsmatrix als auch die Anzahl der Objekte pro Cluster fuzzy sind (Frank, 2002; Höppner u. a., 1997).

2.7 Informationsgewinnung

In Kapitel 2.6 wurde die der Applikation zu Grunde liegende Struktur vorgestellt: das Profil. Die vorgestellten Betrachtungen und Beispiele basierten auf einem schon existenten Datenstamm. Wie die entsprechende Datenstruktur aussehen muss, wurde bereits geklärt: ein Vektor, der auf ganze Zahlen abgebildete Informationen enthält.

Eine wichtige Frage, die man sich nun stellen muss, ist folgende: Wie wird das Profil mit Informationen gefüllt? Dieses Kapitel befasst sich mit der Beantwortung dieser Frage und enthält alle notwendigen Überlegungen, die zu einer wie auch immer gearteten Entscheidung geführt haben.

2.7.1 Grundlegende Überlegungen

In dieser Arbeit soll ein Softwaresystem entwickelt werden, welches im Kern aus mehreren Agenten besteht, die miteinander in Interaktion stehen. Wie in Kapitel 1.4 und Kapitel 1.3 erwähnt, steht jedem Benutzer ein Computer zur Verfügung. Auf diesem Rechner befindet sich die in Kapitel 1.3 genannte Applikation in Ausführung. Diese Applikation hat im Kern einen der oben genannten Agenten. Wenn man die beiden oben genannten Kapitel rekapituliert, werden mehrere Forderungen an die Eigenschaften der Agenten deutlich:

1. Jeder Agent muss in der virtuellen Sozialität den inzidenten realen Benutzer repräsentieren.
2. Jeder Agent muss so weit wie Möglich im Sinne des inzidenten Benutzers handeln.

Diese Forderungen implizieren, dass jeder Agent den inzidenten Benutzer so gut wie möglich „kennen“ muss, um diesen weitestgehend mit der Erledigung seiner Aufgaben wie zum Beispiel der Tischreservierung zufrieden zu stellen. Dazu ist es nötig, dass der Agent Informationen über die individuellen Ausprägungen seines Benutzers in Form des Profil erhält. Zwei Varianten, die zur Gewinnung der Informationen führen, sollen beispielhaft vorgestellt werden.

[11] Je „schärfer“ die Abbruchbedingung ist, desto mehr Zyklen sind notwendig

2.7.2 Beispiele der Informationsgewinnung

Es ist also notwendig, Daten des Nutzers zu kennen, zu sammeln, zu erweitern und auszuwerten. Genutzt werden Varianten der Informationsgewinnung zum Beispiel bei GMX oder dem Online-Book-Shop Amazon[12]. Man bezeichnet diese Form der personalisierten Datenverarbeitung als Profilverarbeitung. Dabei wird versucht, durch Nutzung etwas über den Nutzer zu erfahren und diesem zuzuordnen, so dass zu einem späterem Zeitpunkt auf dieses Profil zurückgegriffen werden kann, sei es zur Erweiterung oder ähnlichem.

Dabei unterscheiden sich die vorgestellten Plattformen in einem Punkt: Bei GMX wird der Nutzer bei Anlage eines Mailaccounts nach seinen Interessen gefragt. Dazu wählt aus einem vorgegebenen Spektrum die Interessen aus, die sich mit seinen decken. Entsprechend der Eintragungen werden Werbungen und Anzeigen, die auf das Profil passen, dem Nutzer bei Gebrauch der Plattform angezeigt. Hierbei kommt es nicht darauf an, den Nutzer mit all seinen Ausprägungen darzustellen, viel mehr benötigt man nur die Interessen, die durch Werbepartner und/oder Inserenten abgedeckt werden.

Amazon geht bei der Profilierung anders vor. Hier wird der Nutzer an Hand seiner Einkäufe und Betrachtungen im Vergleich zu vorhandenen Profilen kategorisiert. So wird dem Nutzer bei Besuch und Gebrauch von Amazon mitgeteilt, was ein hoher Prozentsatz der anderen Nutzer ebenfalls bei Kauf eines Buches gekauft beziehungsweise bei Ansicht eines Buches angesehen hat. Das geht so gar so weit, das spartenübergreifend verglichen wird. Dem Nutzer werden nicht nur Bücher, sondern auch andere Medien vorgeschlagen. Und je länger ein Nutzer die Plattform nutzt, desto besser „lernt“ Amazon den Nutzer kennen (sprich: desto höher die „Trefferquote“ der Vorschläge).

Neben den oben genannten Varianten der Profilierung gibt es noch die der Umfrage. Bei der Umfrage geht es, ähnlich der GMX-Platform, um die Auswahl von Interessen aus einem vorgegebenen Spektrum. Ergänzt wird jede Auswahl noch um eine Art Ausprägung des jeweiligen Interesses beispielsweise mit der Ausprägung „sehr gern“, „sehr schlecht“ etc. Diese Art der Profilierung geht demnach mehr auf den Nutzer ein, denn dadurch ist nicht nur das generelle Interesse, sondern auch gleich die Auskunft darüber vorhanden, wie sehr der Nutzer zu etwas neigt.

Die Nutzung beziehungsweise Anwendung von Profilen wird als Personalisierung bezeichnet, welche sich unter anderem ausdrücken kann durch

- Unterbreitung von individualisierten Angeboten. Dabei sind die Inhalte des Angebotes auf den Nutzer weitestgehend abgestimmt. Dem Nutzer werden somit Dienstleistungen und Produkte offeriert, für die dieser aller Wahrscheinlichkeit nach auch Verwendung findet.

[12] Diese beiden Plattformen sollen hier stellvertretend für andere (semi-)kommerzielle Plattformen stehen.

- Persönliche Anrede. Dabei wird der Nutzer, wie zum Beispiel bei den Plattformen GMX und Amazon, direkt bei Betreten mit dem Vor- beziehungsweise dem vollen Namen angesprochen.
- Interaktive, individuelle Beratung und Unterstützung. Hier werden die vergangenen Nutzungsvorgänge und Datenbestände analysiert und genutzt.
- Individuelles Marketing.
- Die Bereitstellung von individuellen Inhalten/Internetseiten der genutzten Plattform.

Das Ziel der Personalisierung ist der Aufbau einer langfristigen, stabilen und vertrauten Beziehung zwischen dem Unternehmen und dem Kunden/Nutzer als auch die Aquise von Neukunden (Babic, 2003). Um dieses Ziel zu erreichen, wird auf Emotionen und Bedürfnisse des Nutzers explizit eingegangen. Persönliche Anrede, individuell zugeschnittene Angebote etc. dienen dazu, dem Nutzer eine exponierte Stellung und somit das Gefühl von Wichtigkeit und Exklusivität zu vermitteln. Und je mehr das Profil mit den Ausprägungen des Nutzers übereinstimmt, desto besser trifft die Individualisierung auf den Nutzer zu. Das führt dazu, dass sich der Nutzer verstanden und so gar geborgen fühlt. Er beginnt, Vertrauen zu dem Unternehmen aufzubauen und sich nach kurzer Zeit diesem verbunden zu fühlen. Das führt wiederum zur Zufriedenheit des Kunden, die Wiederum zur Steigerung des Unternehmenserfolges führt.

Def.: Unter einem Kundenprofil versteht man eine Menge von Informationen, die direkt einem einzelnen Kunden zugeordnet werden können. Diese Informationen charakterisieren den Kunden hinsichtlich seines Verhaltens, seiner Person und seiner Präferenzen.

Um an die notwendigen Informationen für ein Kundenprofil zu gelangen, gibt es zwei Möglichkeiten:

- Explizite Informationsgewinnung
- Implizite Informationsgewinnung

2.7.2.1 Explizite Informationsgewinnung

Unter expliziter Informationsgewinnung kann das direkte Erfragen von Information verstanden werden. Dabei ist dem Nutzer direkt bewusst, dass er Informationen über sich preisgibt. Beispiele sind Umfragen oder ähnliches.

2.7.2.2 Implizite Informationsgewinnung

Implizite Informationsgewinnung meint den Gewinn von Information durch Analyse des Verhaltens eines Nutzers. Dabei ist dem Nutzer nicht unbedingt bekannt, dass diese Form der Informationsgewinnung auf ihn angewandt wird. Als Beispiel dient hier die Amazon-Plattform. Hier wird aus den Einkaufs- und Betrachtungsgewohnheiten ein möglicher Interessensschwerpunkt gefolgert.

2.7.3 Customer Profile Exchange

Der Standard *Customer Profile Exchange* (CPEX) wurde von diversen, sich zusammengeschlossenen Unternehmen, darunter Macromedia, Calico, net.Genesis, DoubleClick Intuit, IBM, Vignette oder Sun, mit dem Ziel entwickelt, einen „offenen Standard“ zu schaffen, der es möglich macht, die über Kunden mit unterschiedlichen Systemen von verschiedenen Geschäftspartner gesammelten Daten zusammenzuführen und gemeinsam zu nutzen, um dem „Bedürfnis des E-Commerce“ entgegenzukommen, eine „einzige, ganzheitliche Sicht auf ihre Kunden“ zu erhalten.

Dieser Standard soll die online und offline gesammelten Daten in einer XML-basierten Datenbank integrieren, die von unterschiedlichen Programmen online und offline benutzt werden kann.

Dabei sollen die Nutzer, über die Daten gesammelt werden, die Möglichkeit besitzen, den Grad an Datenschutz bestimmen zu können, den sie haben wollen (Rötzer, 1999; CP Exchange 2005).

2.7.4 Platform for Privacy Preferences Project

Das *Platform for Privacy Preferences Project*(P3P) stellt ein Protokoll dar, über das maschinenlesbare Datenschutzerklärungen ausgetauscht werden. Kern dieses Protokolls ist eine standardisierte Liste von Multiple-Choice-Fragen (etwa zu Cookies oder zur Auswertung von Logfiles). Die Antworten ergeben, übersetzt in XML, eine Datenschutzerklärung, die entsprechende Software in die Sprache des jeweiligen Anwenders übersetzen kann.

Das bedeutet, das Webseiten persönliche Informationen mittels Cookies sammeln. Die Betreiber dieser Webseiten geben in ihren *privacy policies* bekannt, wie sie mit den Informationen der Nutzer umgehen. Das hat einen erhöhten Informationsbedarf der Nutzer über eben jene *policies* zur Folge. Das P3P ermöglicht nun die automatische Verarbeitung dieser Informationen. Zu dem können diese Informationen als Entscheidungsgrundlage über den Austausch persönlicher Daten sein (P3P 2005; Schirru, 2004).

2.8 Verteilte Systeme

Wenn man sich an dieser Stelle die Problemstellung und die Zielsetzung (Kapitel 1.2 & 1.3) vor Augen führt, werden mehrere Dinge deutlich. Dem Benutzer stehen mehrere Dienste zur Verfügung, die er an seinem Computer nutzen kann. Dabei kann er mit mehrere anderen Benutzern via Computer interagieren, sei es direkt (Essen, Sport etc.) oder indirekt (Chat etc.).

Der dem Benutzer zugehörige Computer befindet sich also in einem Netzwerk aus mehreren Computern. Da die Organisation der direkten Aktivitäten auf die Rechner der Benutzer ausgelagert worden ist, spricht man hier auch von einem Verteiltem System. Im folgenden wird kurz erläutert, was ein Verteiltes System eigentlich ist.

Verteiltes System, das bedeutet die Aufteilung von Betriebsmitteln, um Engpässe zu vermeiden und das Ausnutzen der dezentralen Natur vieler Anwendungen. Das bedeutet, dass die Prozessoren und anderen Betriebsmittel direkt beim Nutzer angesiedelt sind.

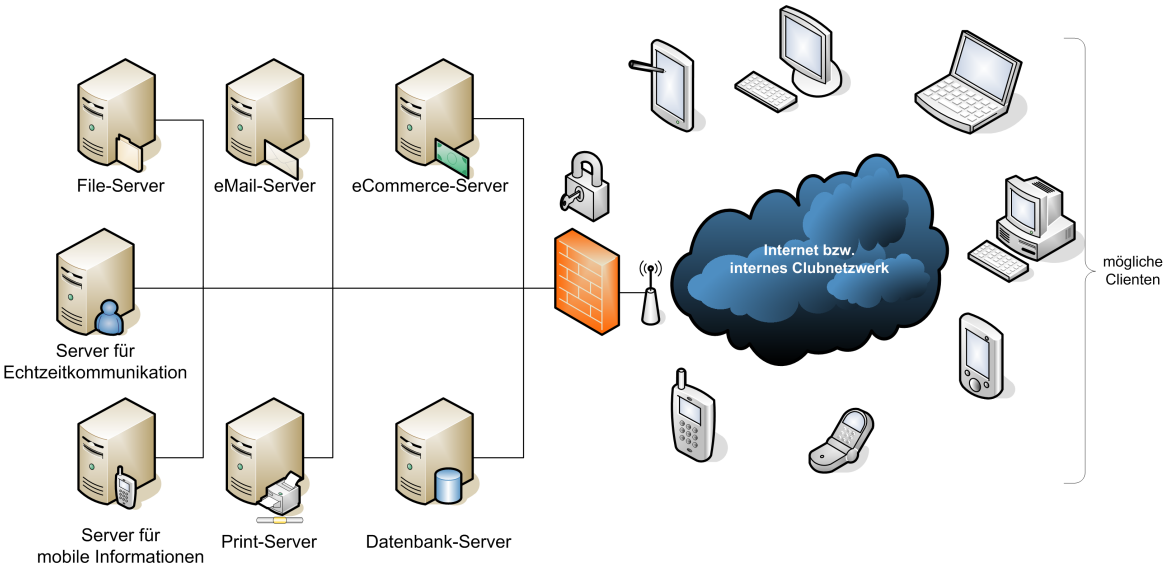

Abbildung 2.9: Das mögliche Clubnetzwerk

Man spricht in diesem Zusammenhang auch von „loser Kopplung“. Die Funktionseinheiten liegen bei dieser Art Kopplung im Gegensatz zur „engen Kopplung“ mehr oder weniger weit auseinander. Das führt dazu, dass zur Kommunikation der lokale Bus entfällt und man hier auf Datenkommunikationsnetze zurückgreifen muss. Diese sind im Vergleich zum lokalen Bussystem eher schmalbandig und langsam.

Um die Probleme von „loser Kopplung“ zu verdecken, realisieren verteilte Systeme ein virtuelles, „eng gekoppeltes“ System auf der Basis von lose gekoppelter Hardware, um Program-

mierern und Anwendern vorzugaukeln, man arbeite auf „eng gekoppelter“ Hardware (Vogt, 2001).

2.8.1 Ziele

Möchte man ein verteiltes System realisieren, sollte man sich folgende Ziele vor Augen halten:

- Man sollte versuchen, eine möglichst hohe Transparenz zu erzeugen. Das bedeutet, dass der Nutzer die räumliche Verteilung nicht bemerkt. Für den Nutzer bedeutet dies, dass er weder um den Ort der Datenspeicherung oder der Prozessausführung wissen muss, noch um die Anzahl der Kopien einer Datei, die auf verschiedenen Knoten ausgeführt werden soll. Ebenfalls sollte er ohne Kenntnis der Quelle auf vorhandene Daten zugreifen können.
- Das System sollte hinsichtlich der Anzahl seiner Knoten leicht erweiterbar sein. In Bezug auf das Beispielszenario bedeutet dies, dass die Anzahl der involvierten Computer variieren kann. Die Variation hat keinerlei Einfluss auf das System. Man spricht hier auch von der Skalierbarkeit eines Systems.
- Zu dem sollte das System flexibel gehalten sein, das heisst, dass einzelne Komponenten leicht verändert beziehungsweise erweitert werden können.
- Das System darf die oben genannten Modifikationen nicht negativ beeinflussen. Es muss hinsichtlich möglich auftretender Fehler tolerant sein und sicher. Man spricht hier von der Zuverlässigkeit des Systems.
- Um den Einsatz eines verteilten Systems zu rechtfertigen, sollte die Leistung des Systems besser sein als die eines lokalen Systems.

2.8.2 Vorteile

Der Einsatz verteilter Systeme bringt, wenn oben genannten Ziele bei der Implementierung berücksichtigt worden sind, einige Vorteile mit sich. So können Betriebsmittel dezentralisiert werden, das bedeutet, dass auf Wunsch Daten und Verarbeitungskapazität „vor Ort"´ gehalten werden können. Dadurch werden verteilte (dezentrale) Anwendungen unmittelbar unterstützt.

Außerdem ist es einfacher, Server zu implementieren und somit einzelne Dienste auszulagern. Es kann also sehr einfach ein Daten-, Funktions- und Anwendungsverbund realisiert werden.

Auch der Ausfall eines oder mehrerer Knoten führt auf Grund der Fehlertoleranz nicht unbedingt unmittelbar zum Stillstand des gesamten Systems. Die Arbeitslast kann auf mehrere Knoten verteilt werden. Man spricht in diesem Zusammenhang auch von einem Last- oder Leistungsverbund. Das setzt jedoch auch voraus, dass bestimmte Anwendungen parallelisiert werden.

Schrittweise Skalierbarkeit ist ebenfalls möglich, wenn alle oben genannten Punkte realisiert worden sind. Dies führt auch dazu, dass man das System einfach anpassen kann.

2.8.3 Probleme

Jedoch bringt die Implementierung verteilter Systeme auch einige Probleme mit sich. Zur Gewährleistung oben genannter Ziele ist komplexe Hard- und Software nötig. Softwaretechnisch erfordert die Realisierung eines verteilten Systems die Entwicklung und Implementierung nebenläufiger Algorithmen. Die Schwierigkeit liegt dabei im Nachweis der Korrektheit (Verifikation) und der Fehlersuche.

Doch auch die an sich vorteilhafte räumliche Verteilung bringt Probleme mit sich. Da auf Grund der Verteilung der herkömmliche Bus der Kommunikation nicht zur Verfügung steht, muss hier auf ein anderes Kommunikationsnetz zurückgegriffen werden. Die Übertragungszeit ist jedoch höher als über den internen Bus. Auch ist durch dieses Medium der Übertragung die Übertragungskapazität beschränkt, was zu Engpässen führen kann.

Auch hat man es bezüglich der räumlichen Verteilung mit einer gewissen Unschärfe zu tun, das heißt, dass an keiner lokalen Stelle eine vollständige, aktuelle Information über das System vorliegt. Betriebsmittel können nur erschwert aufgefunden werden.

In Hinblick auf die Anforderungen an Betriebssysteme lässt sich sagen, dass die Synchronisation von Prozessen und auch die Deadlockbehandlung zu einem größeren Problem werden auf Grund der räumlichen Verteilung. Ebenfalls ist die Systemsicherheit schwieriger durchzusetzen, da kritische Systemkomponenten nicht an zentraler Stelle geschützt werden können. Auch stellen die Übertragunsstrecken potentielle Angriffspunkte dar.

2.9 Software Agenten

In den vorangegangenen Kapiteln wurden Zielsetzung (Kapitel 1.3) und Problemstellung (Kapitel 1.2) näher betrachtet und viele der semantisch expliziten und impliziten Gegenstände betrachtet. Hauptgegenstand der beiden Kapitel war der Kontext der dezentralen Clusteranalyse. Welche Anforderungen an ein derartiges System vorhanden sind, woraus die Cluster bestehen (könnten), die Konfiguration derselbigen und die nähere Betrachtung der Clusterelemente sind neben einigen anderen Dingen in der Analyse schon behandelt worden. Ein Aspekt, der jedoch noch fehlt, ist die Dezentralisierung.

Hält man sich an dieser Stelle einmal die Eigenschaften vor Augen, die die zu entwickelnde Applikation laut den oben genannten Kapiteln mit sich bringen soll:

- Handlung
  - autark und
  - im Sinne des inzidenten Benutzers,
- Interaktion mit anderen Applikationen,
- Verwaltung der Profildaten und
- Clusteranalyse

Wie in der Problemstellung (Kapitel 1.2) erwähnt, agiert die Software im Sinne eines Agenten. Man bezeichnet diesen Softwaretypus auch als Softwareagenten. Diese verfügen im Allgemeinen über eine gewisse Menge an Eigenschaften, den man die oben genannten Anforderungen durchaus zuordnen kann. Bevor jedoch auf diese Eigenschaften eingegangen wird, soll die Agententechnologie an sich kurz vorgestellt werden.

2.9.1 Einführung in die Agententechnologie

Der Bekanntheitsgrad von Softwareagenten steigt seit einigen Jahren stetig an. Dabei beschränkt sich dieser Grad nicht nur auf den Bereich der Informatik. Die Dynamik dieses Gebietes führt jedoch zu Problemen:

- Viele Begriffe sind noch nicht entgültig fixiert.
- Ebenso gibt es für die Frage nach dem technologischem Unterbau noch einen großen Spielraum hinsichtlich der Antwort.

Dem gegenüber steht jedoch, dass es schon zahlreiche erfolgreiche Systeme gibt. Auch die Grundlagen dieses Gebietes sind weitestgehend erforscht und fundiert. Doch was ist die Motivation hinter dieser „neuen“ Technologie?

Die steigende Verbreitung von Computersystemen und die Tatsache, dass der Umgang mit ihnen nicht mehr nur den Experten vorbehalten ist, sorgt für eine neue Sicht auf die Softwareentwicklung. Es gilt nicht mehr Probleme durch möglich viele Varianten zu lösen, sondern dem Benutzer nur noch eine Möglichkeit der Problemlösung bereit zu stellen. Die Erwartung an Software ist gestiegen.

Was bedeutet „Agent“ im Zusammenhang mit Software eigentlich?

Def.: „Ein Softwareagent ist ein längerfristig arbeitendes Programm, dessen Arbeit als eigenständiges Erledigen von Aufträgen oder Verfolgen von Zielen in Interaktion mit einer Umwelt beschrieben werden kann.“(Görz u. a., 2003)

An dieser Stelle kann man eine Unterscheidung treffen: Handelt es sich bei dem Agenten um eine Applikation, die einzigartig in dem System ist oder gibt es mehrere Agenten, die eventuell kooperieren? Letzteres wird auch als Multiagentensystem bezeichnet. Multiagentensysteme bestehen demnach aus mehreren Agenten. Dabei gibt es mehrere Formen der Kooperation.

So kann es sein, das mehrere Agenten in Bezug auf eine Aufgabe „aufeinander treffen“. Um diese Aufgabe gemeinsam zu lösen, tritt man in Verhandlung miteinander und löst die Aufgabe gemeinsam. Jeder der involvierten Agenten ist für die Lösung einer anderen Aufgabe zuständig und ist somit Experte auf seinem Gebiet.

Ein anderer Fall ist der, dass es zwar wiederum eine Aufgabe zu lösen gilt und mehrere Agenten sich in einem System befinden, doch arbeiten sie diesmal nicht gemeinsam an der Lösung einer Aufgabe, sondern verfolgen individuelle Ziele.

Zu guter Letzt ein Szenario, welches immer populärer wird. Mehrere Agenten arbeiten in einem System zusammen an der Lösung einer Aufgabe. Jedoch ist die Intelligenz jedes einzelnen Agenten relativ einfach im Vergleich zu den quasikomplexen System der anderen beiden Fälle. Das interessante jedoch an diesem Fall ist, dass die Betrachtung der Intelligenz bzw. des Verhaltens der Summe dieser Agenten doch eine relativ komplexe Verhaltensstruktur und Intelligenz aufweisen kann. Der einzelne ist sozusagen wertlos, in der Gesamtheit sind sie jedoch sehr wertvoll. Man spricht in diesem Zusammenhang auch von „Schwarm-Intelligenz“. Ihr zu Grunde liegt die Erforschung der Verhaltensweise von Beispielen aus der Tierwelt, seien es Bienenvölker oder Ameisen etc. (Kennedy u. a., 2001).

Wir sehen also, dass Mulitagentensysteme ein vielversprechendes Mittel zur Abbildung und Erforschung sozialer Handlungen darstellt. Warum das so ist, soll nachfolgend erläutert werden.

In Bezug auf Softwareagenten gibt es eine Reihe von Eigenschaften, die in Zusammenhang mit der Erforschung dieser Technologie mit diesem Begriff einhergehen (siehe Tabelle 2.5). Jedoch ist der Grad der Erfüllung dieser Eigenschaften kontextsensitiv. Ergo muss nicht jeder Agent zu 100% über jede Eigenschaft verfügen.

Eigenschaft	**Erklärung**
Andauernde Verfügbarkeit/Aktivität	Im Grunde ist damit gemeint, dass der Agent über einen längeren Zeitraum „ansprechbar“ ist. Dadurch ist gewährleistet, dass dieser eigenständig aktiv werden kann.
Interaktion mit der Umwelt	Der Agent nimmt Informationen aus seiner Umwelt auf und verarbeitet diese. Er interagiert mit ihr, um sie im Sinne des „Auftrags“ zu beeinflussen.
Situiertheit	Betrifft auch die Interaktion mit der Umwelt. Es wird jedoch hervorgehoben, dass komplexes Verhalten auch als Resultat von direkter Reaktion auf Umwelteinflüsse erzeugbar ist. Dies führt zu einer einfacheren Architektur.
Autonomie im Handeln	Der Benutzer eines Softwareagenten muss diesen nicht in Bezug auf Handlungsweise vollständig kontrollieren und steuern. Vielmehr handeln sie „im Sinne der Auftraggeber“. Diese Handlung kann nach sehr komplexen Methoden erfolgen.
Reaktivität	Die Eigenschaft zur Interaktion mit der Umwelt.
Reaktivitätes Verhalten	Bedeutet unmittelbares Reagieren auf Umwelterereignisse.
Zielgerichtetheit	Das heisst, dass Agenten- auch über einen längeren Zeitraum hinweg - ein dem Ziel angepasstes Handeln an den Tag legen.
Pro-Aktivität	Im Grunde bedeutet es Zielgerichtetheit, jedoch betont man in diesem Kontext die Eigeninitiative des Agenten.
Deliberatives Verhalten	Bedeutet die Auswahl von Zielen bzw. Aufträgen nach Vorgabe. Steht sozusagen im Kontrast zu reaktivem Verhalten, welches unmittelbar erfolgt.
Intelligenz	Analog zur Umschreibung der Künstlichen Intelligenz: Das an den Tag gelegte Verhalten gilt auch beim Menschen als intelligent.

Rationalität	Es wird immer eine sinnvolle Entscheidung getroffen. Selbst bei eingeschränkten eigenen Resourcen erfolgt die Entscheidungsfindung in angemessener Zeit.
Komplexität	Beschreibt die Komplexität der einzelnen Komponenten eines Agenten.
(Persistente) Zustände	„Gedächtnis“ eines Agenten in Form eines Zustandes. Es können Informationen bzgl. vergangener Ereignisse, Ziele oder Pläne über längere Zeiträume gespeichert werden.
Lernfähigkeit	Anpassung des Verhaltens an die Umwelt.
Mobilität	Eigenständige Migrierung auf andere Plattformen.
Kooperation	Zusammenarbeit mit Menschen und/oder anderen Agenten.
Wohlwollen	Handlung im Sinne des Auftraggebers. Dabei kann auch ein anderer Agent in die Rolle des Auftraggebers schlüpfen.
Soziales Verhalten	Kooperation unter Beachtung vorgegebener Normen und Regeln, Fähigkeit der Gruppen- oder Koalitionsbildung nach dem Vorbild sozialer Strukturen der Menschen.
Emotionales Verhalten	Emotionen sollen Verhalten steuern und Interaktion mit dem Menschen verbessern.
Glaubwürdigkeit	Visualisierung muss mit Verhalten und Interaktion übereinstimmen.

Tabelle 2.5: Eigenschaften von Softwareagenten

Auch wenn ein Agent nicht alle Eigenschaften notwendigerweise mit sich bringen muss, so gibt es doch ein paar Eigenschaften, die ein Agent immer erfüllt. Dies sind die Eigenschaften

- Andauernde Verfügbarkeit/Aktivität,
- Autonomes Handel und
- Interaktion in einer Umwelt, wobei dies reaktiv oder deliberativ sein kann.

In Anbetracht dieser Eigenschaften lässt sich ein Zyklus ausmachen, der immer wieder durchlaufen wird.

3-Phasen-Zyklus

Der Zyklus, der immer wieder von einem Agenten durchlaufen wird, kann in drei Phasen unterteilt werden:

Phase	Beschreibung
1. Informationsaufnahme	Der Agent nimmt Informationen aus der Umwelt auf und beobachtet die Wirkungen seiner Handlungen.
2. Wissensverarbeitung und Entscheidung	Es wird das vorhandene Wissen mit Hilfe der Informationen aus Phase 1 aktualisiert. Die aktuelle Situation, neue Aufträge und der Fortschritt werden analysiert und bewertet. Der Agent trifft Entscheidungen bezüglich unmittelbarer und zukünftiger Aktionen.
3. Aktionsausführung	Die Ergebnisse aus Phase 2 werden in die Tat umgesetzt.

Tabelle 2.6: Der 3-Phasen-Zyklus

Diese drei Phasen können sowohl parallel als auch sequentiell abgearbeitet werden. Üblicherweise erfolgt die Abarbeitung jedoch sequentiell, da die Phasen in kausaler Abhängigkeit zueinander stehen. Überschneidungen können folglich zu Konfusionen führen, was ein entsprechendes Zeitmanagement und die Verwaltung eventueller Zustände nötig machen.

Unter Berücksichtigung der bisher vorgestellten Eigenschaften und des 3-Phasen-Zyklus lassen sich (komplexere) Softwareagenten in mehrere Grundkomponenten einteilen (siehe Tabelle 2.7) (Görz u. a., 2003).

2.9.2 Technische Problemstellungen

Wenden wir uns für einen kurzen Moment der Softwaretechnik im Allgemeinen zu und stellen und die Frage nach dem Gegenstand die Disziplin der Computerwissenschaften. Dieser kann wie folgt formuliert werden:

Def.: Softwaretechnik ist die Disziplin der Informatik, die sich mit der inhaltlichen Beschreibung von Problemen und ihrer maschinensprachlichen Formulierung beschäftigt.

Wer sich mit softwaretechnischer Problemlösung und Realisierung beschäftigt hat, wird zustimmen, dass eine Umstellung der eingesetzten Technologien einen erheblichen Aufwand

Komponente	Aufgabe
1. Umweltkopplung	Realisiert die ersten beiden Phasen des 3-Phasen-Zyklus (siehe Tabelle 2.6).
2. Steuerung & Management	Stellt die notwendigen Middleware-Funktionen zur Verfügung und enthält alle für den Programmablauf notwendigen Funktionen - das Betriebssystem.
3. Fähigkeiten 4. Umweltmodell 5. Entscheidungkomponente	Zusammen stellen diese Komponenten die „Intelligenz“ des Agenten dar und realisieren die letzte Phase des 3-Phasen-Zyklus (siehe Tabelle 2.6). Hier finden Methoden aus dem Bereich der Künstlichen Intelligenz Anwendung.

Tabelle 2.7: Die Basiskomponenten von Softwareagenten

mit sich bringt. Zwingende Umstände und/oder erhebliche Vorteile sind meist die einzigen Gründe für einen Wechsel.

In Bezug auf Agententechnik ist die Diskussion über mögliche Ausmaße der Technologie auf Grund des geringen Alters noch nicht beendet. Die Interdisziplinarität der Agententechnologie impliziert, dass mögliche „Wunschlisten“ sehr umfangreich werden können, denkt man nur an die involvierten Bereiche

- Informatik mit

  Verteilten System,

  Computernetzen,

  Logischer Programmierung und

  Objektorientierter Programmierung
- Management
- Datensicherheit
- Datenschutz
- Soziologie
- Philosophie
- Psychologie
- Ökonomie etc.

All diese Bereiche zu durchleuchten würde den Rahmen einer Diplomarbeit bei weitem übertreffen. Doch den Bereich der Verteilten Systeme, der in Kapitel 2.8 allgemein erläutert wurde, kann und sollte man an dieser Stelle näher betrachten, da in diesem Bereich schon einige Möglichkeiten der Unterstützung bei der Entwicklung und Realisierung von Agentensystemen vorhanden sind.

Für die Zusammenarbeit von Programmen auf unterschiedlichen Systemen, wie sie in Kapitel 2.8 beispielhaft gezeigt wurden, ist ein sehr komplexes System von Protokollen und Diensten notwendig. Diese Protokolle und Dienste regeln einen aufeinander abgestimmten Ablauf. Diese Strukturierung ist auch bekannt als „Schichten-Modell“ und kann in folgende Schichten unterteilt werden:

Schicht	Name
5	Verarbeitungsschicht
4	Transportschicht
3	Vermittlungsschicht
2	Sicherungsschicht
1	Bitübertragungsschicht

Tabelle 2.8: Das Schichtenmodell

Dieses Modell kann als bekannt vorausgesetzt werden, da es in diversen Lehrbüchern eingehend behandelt wurde und zu dem auch nicht Gegenstand dieser Arbeit ist. Wer dennoch mehr wissen möchte, dem sei (Vogt, 2001) an Herz gelegt.

Diese fünf Schichten sind für die Arbeit (intelligenter) Softwareagenten Voraussetzung. Es gibt aber weitere Protokolle, die eine Zusammenarbeit zwischen den Agenten regeln. Diese liegen hinsichtlich der in Tabelle 2.8 aufgeführten Schichtenarchitektur in der Verarbeitungsschicht. Und wenn man sich schon mit der Kommunikation der Agenten untereinander beschäftigt, muss man sich über folgendes im Klaren sein: Damit Agenten überhaupt untereinander kommunizieren können, müssen sich diese „kennen“. Auch hier gibt es Möglichkeiten der Registrierung von Agenten und ihren Diensten. Sogenannte *Broker* verwalten die registrierten Dienste und vermitteln zwischen Angebot und Nachfrage. Dabei ist lediglich eine Beschreibung der Dienste vorhanden. Die Details der Implementation bleiben verborgen.

Elementar sind für die Entwicklung von Agentensystemen die Modellierung der Umwelt und die Kommunikation, sei es zwischen Agenten, Agent und Benutzer oder was auch immer. Diese beiden Dinge werden in den folgenden Kapiteln näher erläutert.

2.9.3 Umwelt

Damit ein Agent sinnvoll in einer Umgebung agieren und interagieren kann, muss diesem ein „Grundverständnis“ derselbigen gegeben werden. Dies muss in einer Form geschehen, die formal eineindeutig ist, da sich der Agent mit anderen über diese Umwelt austauscht. Natürlichsprachliche Spezifikation - möglichst noch via Spracheingabe - ist hier nicht angebracht, da der Erfolg der Analyse kleiner 100% ist. Somit wäre die geforderte Eindeutigkeit nicht gewährleistet. Wie kann notwendiges Wissen an dieser Stelle vermittelt werden? An dieser Stelle sollte man sich erst einmal drüber im klaren sein, worum es eigentlich genau geht.

Man nehme einmal an, dass man uns über das Thema „Auto“ unterhält. Wie ist der Verlauf unserer Unterhaltung? Was der genaue Gegenstand? Erzählen wir sofort bei dem Thema, welches Auto wir fahren? Welcher Motor vorhanden ist? Oder beginnen wir das Gespräch mit der Erklärung der Optimierung der Einspritzpumpe für variable Geschwindigkeiten? Verkleinern wir den Fokus ein bisschen. Wir befinden uns in einer Unterhaltung. Automatisch treffen wir in der zwischenmenschlichen Kommunikation Annahmen über den Kenntnisstand des Gesprächspartners zu diesem Thema. Doch woher wissen wir, dass diese Annahmen auch zutreffen? Dieses Vorgehen trifft auf alle Bereiche des Wissensaustausches zu. Man benutzt unterschiedliche Wortschätze, um Inhalte unterschiedlicher Bereiche korrekt zu beschreiben. Dabei kann es sehr fachorientiert unter Spezialisten oder aber auch relativ leicht verständlich sein. Man bedient sich hier eines Vokabulars, einer Liste der Elementbeziehungen, die ein Thema charakterisieren.

Doch ist das Vokabular nicht das einzige Charakteristikum eines Bereichs. Bereiche verfügen über implizite Strukturen, welche die Beziehungen der Elemente definieren. Diese Beziehungen sind es, die von „Außenseitern“ oft nur schwer verstanden werden. Diese beiden Punkte können zusammengefasst werden.

Def.: Unter einem „Weltbild“ versteht man eine Menge von Elementbeziehungen - das Vokabular - und die Strukturen, welche die Beziehungen der Elemente definieren.

Es gilt also, in der Agentenorientierten Softwareentwicklung ein in sich geschlossenes, eindeutiges Weltbild zu definieren, so dass jede involvierte Instanz widerspruchsfrei über dieses kommunizieren kann. Somit müssen zwei Dinge geklärt werden:

1. Wie wird Wissen dargestellt?
2. Wie wird es kommuniziert?

2.9.4 Wissenspräsentation und Kommunikation

Wenn wir uns Gedanken darüber machen wollen, wie Wissen repräsentiert werden kann, so lässt sich dies am besten an einem Beispiel tun. Nehmen wir an, ein Agent soll in der Welt von Hemden aktiv werden. Dieser Agent benötigt eine Inhaltsstruktur, mit der sich die Welt der Hemden reflektieren lässt - das Vokabular. Diese Vokabular umfasst beispielsweise

- lange Ärmel,
- kurze Ärmel,
- Stehkragen,
- Spitzkragen,
- Baumwollstoff etc.

Die Struktur dieser Welt besteht unter anderem aus

- Ärmeln,
- Kragenart,
- Stoffart etc.

Man kann, wenn man den Focus auf den Bereich auf Kleidungsstücke verallgemeinert, leicht erkennen, dass an dieser Stelle Konzepte wie Objektorientierung und Vererbung die Lücke schließen (A.K.Caglayan und C.G.Harrison, 1998). Es gibt jedoch auch eine Form der agentenspezifischen Wissensdarstellung, wie wir später sehen werden. Am sinnvollsten ist es jedoch, die Evolution der Agentensprachen zu betrachten.

Evolution

Die Art und Weise, wie Agenten miteinander kommunizieren können (und sollen) wurde häufig untersucht (Görz u. a., 2003). Entsprechend vielfältig sind Definitionen und Spezifikationen von Kommunikationssprachen. (Wooldridge, 2002) hat verschiedene Sprachen untersucht. Dabei legte er den Schwerpunkt auf die Sprachen, die nicht objektorientiert sind. Er kam zu dem Schluss, dass objektorientierte Programmiersprachen eine Art Vorgänger von Agentensprachen sind, da das zu Grunde liegende Konzept sehr ähnlich ist. Jedoch geht das Konzept der Agentensprachen noch ein wenig darüber hinaus, denn neben dem Weltbild, welches mittels Objektorientierung modelliert werden kann, verfügen Agentensprachen über eine Semantik, die eine eindeutige Kommunikation erlaubt. An dieser Stelle soll ein kurzer historischer Abriss der Entwicklung von Agentensprachen bis hin zu einem weit verbreitetem Standard aufgezeigt werden.

Knowledge Sharing Effort

Die erwähnte Sprachvielfalt ist nicht nur ein Segen, sondern auch ein Hindernis für deren Einsatz, da man eine entsprechende Kurzlebigkeit der ausgewählten Sprache fürchtete. Aus diesem Grunde wurde ein Forschungprogramm gefördert, das unter dem Namen *Knowledge Sharing Effort* (KSE) bekannt ist.

Man beschäftigt sich innerhalb dieses Forschungsprogrammes mit der Entwicklung von Konventionen zu gemeinsamen Nutzung von Wissensdatenbanken und Wissenssystemen. Dabei ist das Ziel die „Definition, Entwicklung und das Testen der Infrastruktur und unterstützender Technologie, um den Teilnehmern zu ermöglichen, größere und allgemeingültigere funktionale Systeme zu errichten, die als Einzelanwendungen nicht erstellt werden könnten“.Das KSE entwickelte zwei ähnliche Sprachen: KIF und KQML. Diese beiden Sprachen sollen nachfolgend kurz vorgestellt werden.

Knowledge Interquery Format

Das *Knowledge Interquery Format* (KIF) ist mit dem Ziel entwickelt worden, mit Hilfe eines allgemeinen Austauschformats eine Sprache zur Entwicklung intelligenter Anwendungen aufzubauen. Der Schwerpunkt dieser Sprache liegt deutlich auf Interoperabilität. Betrachtet man sich KIF näher, so ist KIF eine Syntax für den Nachrichtenaustausch zwischen Agenten. Vom Aussehen her ähnelt es LISP. Intention war jedoch nicht, eine weitere Sprache zu entwickeln, sondern vielmehr ein allgemeines Austauschformat zwischen verschiedenen Agentensprachen zur Verfügung zu stellen. KIF beschäftigt sich demnach also mit der Darstellung von Wissen (A.K.Caglayan und C.G.Harrison, 1998).

Knowledge Query and Manipulation Language

Die *Knowledge Query and Manipulation Language* (KQML) setzt quasi auf KIF auf, den man definierte hier eine Syntax für die Nachrichtenübermittlung zwischen Agenten. Es liegt hier also schon ein gewisser Befehlsvorrat zu Grunde. Somit ist das Wesen von KQML im Gegensatz zu KIF nicht die Wissensdarstellung, sondern Nachrichtenformate und Nachrichtenverwaltungsprotokolle[13]. Dabei sind jedoch nur die Protokolle, nicht aber die verwendete Sprache spezifiziert. In KQML ist eine große Anzahl von Nachrichten definiert, welche dazu dienen, bestimmte Aktionen implizit auszuführen. Der Nachteil von KQML ist, dass Agenten, die via KQML kommunizieren möchten, nicht direkt miteinander in Kontakt treten, sondern einen sogenannten KQML-Facilitator oder Vermittler benötigen. Auch die Kompatibilität

[13] KQML benutzt einen anderen Standard der Kommunikation in verteilten Systemen: CORBA. Dieser Standard wurde von der Agent Working Group der Object Management Group (OMG) entwickelt. CORBA umfasst dabei Kommunikationsdienste, Ereignisübermittlung, Verzeichnis- und Namensdienste. CORBA erlaubt - durch entsprechende Implementationen - die verteilte Arbeit von Objekten ohne Kenntnis der jeweiligen Plattform. In wie fern andere Dienste wie SOAP oder WebServices CORBAs Position streitig machen, soll hier nicht untersucht werden.

zwischen „KQML-sprechenden“ Agenten ist nicht gewährleistet, da dies stark von den verwendeten Nachrichten abhängt. Dennoch ist KQML die am häufigsten verwendete Technik (Bigus und Bigus, 2001).

Foundation for Intelligent Physical Agents

1995 begann die *Foundation for Intelligent Physical Agents (FIPA)* damit, Standards für Agenten Systeme zu entwickeln. Einer dieser Standards basiert auf der KQML, erweitert und verändert diese jedoch: die *ACL - Die Agent Communication Language*.

1997 wurde der erste Standard veröffentlicht: der FIPA-97 Standard. Die behandelten Hauptgebiete waren

- Verwaltung von Agenten,
- Kommunikation zwischen Agenten und
- Integration von Agentensoftware.

Da dieser jedoch noch einige Schwächen hatte und Java RMI als auch HTTP in ihrer Bedeutung aufstiegen, sah man darin einen der Hauptfaktoren zur Entwicklung einer neuen, technologieunabhängigen Spezifikation.

Das Ergebnis ist die FIPA2000-Spezifikation. In ihr wurde eine abstrakte Architektur definiert, was wiederum alternative Implementierungen ermöglicht. Es wird im Grunde nur ein Referenzmodell zur Agentenverwaltung definiert (siehe Abbildung 2.10).

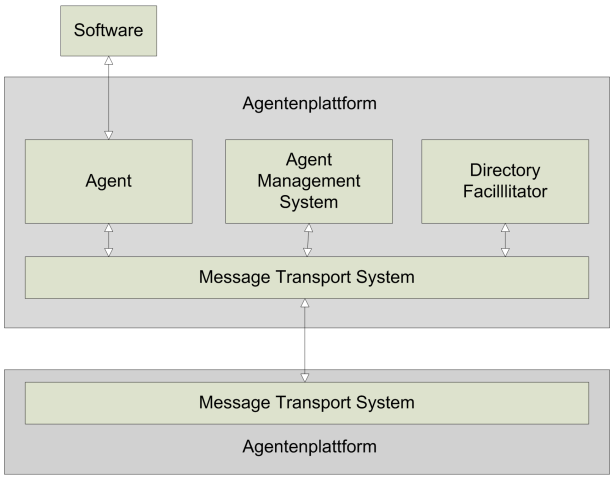

Abbildung 2.10: Die FIPA-Agentenplattform

Eine Agentenplattform stellt ein System zur Verwaltung von Agenten , einen Verzeichnis-Facilitator und ein System zum Transport von Nachrichten zur Verfügung. Das System zur Verwaltung von Agenten - das **Agenten Management System (AMS)** - regelt die Lebenszyklen der Agenten. Der Verzeichnis-Facilitator fungiert als Suchdienst, ähnlich den Gelben Seiten. Das System zum Transport von Nachrichten - das **Message Transport System (MTS)** - erlaubt sowohl interne Kommunikation zwischen Agenten als auch externe Kommunikation mit anderen FIPA-konformen Plattformen (Bigus und Bigus, 2001).

Die FIPA ACL ist im Grunde wie die KQML. Es wird ein fester Satz von Nachrichten zur Verfügung gestellt. Der Unterschied ist jedoch, dass die Nachrichten der FIPA ACL kategorisiert sind, die der KQML nicht (Wooldridge, 2002).

Die in Kapitel 2.9.2 genannten Protokolle und Dienste entsprechen den Agentensprachen und werden unter dem Begriff „ Middleware“ zusammengefasst.

2.9.5 Exkurs Sozionik

Multiagentensysteme werden neuerdings dazu verwendet, soziale Gefüge und das Zustandekommen von Kommunikation bzw. ihrer Ausprägung zu untersuche. Dazu wurde an der Technischen Universität Hamburg-Harburg ein DFG Forschungschwerpunktprojekt initiiert, welches als Bindeglied zwischen dem Faktor Mensch und der Informatik anzusehen ist.

Prof. Dr. Thomas Malsch ist Koordinator eines Schwerpunktprojektes des DFG namens „Sozionik“. Dieses SPP ist ein interdisziplinäres Forschungsprojekt, welches sich zwischen Soziologie und Informatik befindet. Das Wesen der Sozionik liegt in der Transdisziplinarität zwischen Soziologie und Informatik, insbesondere der Verteilten Künstlichen Intelligenz und nutzt Anregungen und Analogien aus der sozialen Welt, um „intelligente“ Technologien zu entwickeln und so technische Probleme nach dem Vorbild sozialer Problembewältigung zu lösen und gleichzeitig soziologische Theoriebildung durch Modellierung und experimentelle Simulation voranzutreiben. Die trotz aller Krisen erstaunliche Wandlungsfähigkeit menschlicher Gesellschaften und deren Stabilitätseigenschaften soll genutzt werden, künstliche Gesellschaften nach diesem Vorbild zu gestalten und darauf aufbauend Problemlösungsmethoden für komplexe, auch hybride Prozesse in Organisationen zu entwickeln (von Lüde u. a., 2004).

Man bedient sich der Multiagentensysteme aus der Verteilten Künstlichen Intelligenz als Werkzeug zur Abbildung sozialer Strukturen. Jeder Agent ist somit Avatar für eine reale Person. Doch wo die MAS der VKI ihre Handlungen nach einem festgelegten Regelsatz vollziehen, wird hier jeder „Dialog“ zwischen zwei oder mehreren Agenten durch ein Team von Soziologen und Psychologen begleitet, die die Entscheidung, welche Agent wie handelt,

bestimmen. Diese Entscheidung wird mit jedem „Dialog“ neu gefällt. Somit ist die Struktur der Handlungen in höchstem Maße stochastisch und nicht deteministisch!

Das Augenmerk der Sozionik bzw. der involvierten Forscher liegt somit nicht auf den Möglichkeiten der Kooperation zwischen Multiagentensystemen, sondern vielmehr auf dem Erstellen der Regeln, denen diese Möglichkeiten unterworfen sind.

Man erhofft sich außerdem, dass die Verbesserung der Technologie in bidirektionaler Wechselwirkung mit ihrem Vorbild steht, dass also auch soziale Strukturen verbessert durch die Ergebnisse sozionischer Forschung werden können. Dabei handelt es sich jedoch in keinster Weise um eine Ausprägung mathematischer Soziologie bzw. ihrer Simulationen, sondern um die interdisziplinäre „sozionische“ Umarbeitung von sozilogischen Theorien zu generischen Softwaretechnologien. Im Unterschied zu generischer Technologie bezieht sich die angesprochene interdisziplinäre Gestaltung auf die praktische Anwendung, das heißt auf die gebrauchsspezifische Anpassung generischer Technik.

2.10 Fazit

Man hat gesehen, was mit der gegebenen Problemstellung und Zielsetzung alles einhergeht und in welchem Kontext man sich bewegen. Da das zu entwickelnde System permanent aktiv ist, kann man auch hier von einer Allgegenwärtigkeit sprechen. Ebenso ist dieses System ein Teil der (Club)umwelt und somit ist die Einordnung in die Gebiete des ubiquitären Computings bzw. der Ambient Intelligence durchaus berechtigt.

Die erwähnten allgemeinen Anforderungen der Informatik sollen auch in der Realiserung ihre Gültigkeit haben und werden dabei immer bedacht. Die datenschutzrechtlichen Aspekte spielen zwar generell eine Rolle, werden jedoch in dieser Arbeit bewusst ausgeklammert, da der Datenschutz nicht Gegenstand dieser Arbeit ist. Eine Erwähnung ist jedoch notwendig, damit dieser Aspekt bei einer möglichen Weiterentwicklung bedacht und berücksichtigt werden kann.

Bei den Anforderungen aus dem Beispielszenario wird in Zukunft der Administrator eine eher untergeordnete Rolle spielen und von daher eher als gegeben vorausgesetzt. Unser Augenmerk liegt auf dem Gast und seinem sozialen Umfeld. Selbiges gilt für die Anwendungsfälle. Die des Administrators sollten bekannt sein, werden aber nicht weiter verwendet. In Bezug auf die Anwendungsfälle des Gastes wird die Möglichkeit der Unternehmungsangebotes seitens des Gastes nicht weiter betrachtet und als eine Erweiterung des mit dieser Arbeit zu entwickelnden Systems außer Acht gelassen. Alle anderen Anwendungsfälle werden bei dem Entwurf und der Realisierung berücksichtigt. Dabei spielen vor allem die Profilanlage beziehungsweise -bearbeitung eine wichtige Rolle, da diese essentiell für die Clusteranalyse sind.

Der Auslöser für die Analyse ist die Annahme einer Einladung. Bezüglich der Anwendungsfälle liegt hier dann der Focus.

Profile werden hinsichtlich der Anwendung als ein (Orts)Vektor von Zahlen betrachtet. Da es sich wie bereits erwähnt jedoch immer um Selbsteinschätzung bei der Profileingabe handelt, ist die Behandlung der Elemente des Vektors als konkrete Zahlen nicht angebracht. Eine adäquate Behandlung erfolgt via Fuzzy-Logik, die in dieser Arbeit realisiert wird. Einhergehend mit der Verwendung vagen Wissens folgt die Verwendung einer entsprechenden Methode zur Bildung von Gruppen. Dies wird mittels des Fuzzy-C-means-Verfahrens erreicht. Eine Gruppenbildung mittels neuronaler Netze ist hinsichtlich der Komplexität, der Lernphase und der Verwendung auf mobilen Endgeräten als nicht praktikabel eingestuft und wird somit nicht weiter berücksichtigt.

Ma hat gesehen, welche Formen der Informationsgewinnung es gibt. In dieser Arbeit wird die direkte Informationsgewinnung verwendet, da indirekte Informationsgewinnung selbst Stoff für mehrere Arbeiten liefert. Von daher wird sie als Erweiterungsmöglichkeit im Ausblick in Form der dynamischen Profilierung erwähnt und nicht näher betrachtet.

Über die verteilten Systeme ist man zu den Agenten gekommen. Es gab einen kurzen Einblick in die Komplexität der Agententechnologie. Dabei wurde deutlich, dass die gezeigte komplexität nur im Teil umgesetzt werden muss. In dieser Arbeit werden folgende Eigenschaften weiterhin ihre Gültigkeit haben und realisiert werden:

- Andauernde Verfügbarkeit/Aktivität
- Interaktion mit der Umwelt
- Autonomie im Handeln
- Reaktivität und Reaktivitätes Verhalten
- Intelligenz
- (Persistente) Zustände
- Kooperation
- Wohlwollen
- Soziales Verhalten
- Glaubwürdigkeit

Auch der erwähnte 3-Phasen-Zyklus soll mit dem folgenden Kapitel umgesetzt werden. Als Kommnunikationsmittel soll der FIPA-Standard verwendet und umgesetzt werden.

Somit verfügt man an dieser Stelle über das geeignete Rüstzeug, um die zu entwicklende Software zu entwerfen.

3 Design und Realsierung

Ausgehend von den Erkenntnissen aus Kapitel 2 soll in diesem Kapitel das gewonnene Wissen konkretisiert und praktisch umgesetzt werden.

So beginnt das Kapitel 3.1 mit der Verknüpfung der Inhalte von Kapitel 2.6.2 und Kapitel 2.9. Es kommt also einer Verschmelzung der mathematischen Erkenntnisse und der Agententechnologie. Hier wird eine Möglichkeit aufgezeigt, ein zentrales Verfahren zu dezentralisieren.

Anschließend werden in Kapitel 3.2 einige Aspekte der Gruppenbildung in Bezug auf Kapitel 1.4 vorgestellt. Diese sind notwendig und müssen in Bezug auf die Clusteranalyse berücksichtigt werden.

Danach geht man über in die theoretische Umsetzung. In Kapitel 3.3 lernt man die interne Struktur des Systems kennen. Dabei werden Lösungen bekannter Probleme an entsprechender Stelle aufgegriffen und umgesetzt.

Zur Überführung der theoretische Umsetzung in die Realität werden in Kapitel 3.4 individuelle Ansichten diskutiert. Dabei geht es vom verwendeten, programmiersprachlichen Paradigma bis hin zur Kommunikationsschnittstelle.

Die Erkenntnisse der praktischen Umsetzung werden abschließend in Kapitel 3.5 vorgestellt.

3.1 Zentrale vs. dezentrale Clusterbildung

(Butrynowski, 2005) stellte in der Studienarbeit einen Ansatz zur zentralen Clusterbildung vor. In dieser Arbeit geht es um die Dezentralisierung dieses Ansatzes.

3.1.1 Unterschiede

Wenn man sich an dieser Stelle die beiden vorgestellten Ansätze der Clusterbildung vor Augen hält, ist eine starke Ähnlichkeit nicht von der Hand zu weisen. Die beiden Formen überschneiden sich sogar in einigen Bereichen (siehe Abbildung 3.1).

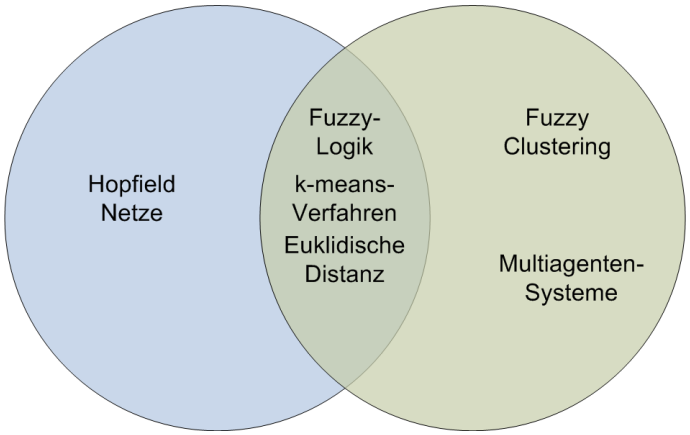

Abbildung 3.1: Zentrale vs. dezentrale Clusteranalyse

Der Unterschied steckt hier im Detail. Während (Butrynowski, 2005) in seiner Arbeit die Euklidische Distanz, Hopfield Netze beziehungsweise Clusteranalyse mittels k-means-Algorithmus verwendet, wird in dieser Arbeit zwar auch mit der Euklidischen Distanz gearbeitet, jedoch erfolgt dann eine Verteilung der Arbeit auf mehrere voneinander unabhängige Systeme. Es gibt bei (Butrynowski, 2005) eine zentrale Komponente, welche die Erstellung einer Sitzordnung vornimmt. Dies ist eine stark passive Form in Bezug auf die benutzerinzidenten Endgeräte.

Die dezentrale Aufgabenverteilung macht eine zentrale Einheit überflüssig und sorgt für aktive Verhandlung der benutzerinzidenten Endgeräte untereinander.

3.1.2 Das Clusteringverfahren

In Bezug auf die zentrale Clusterbildung wurden herkömmliche Verfahren der Clusteranalyse mit der infrastrukturellen Eigenschaft des in (Butrynowski, 2005) vorgestellten Ansatzes kom-

biniert. Es gibt dort eine zentrale Einheit - einen Server - auf dem die Profile sämtlicher Gäste abgelegt werden. Sollte es nun zu Tisch gehen, kam das *k-means*-Clusteringverfahren zum Einsatz. Es wurden k Clusterzentren zufällig bestimmt. Anschließend wurden die verbleibenden Profile den Gruppen zugewiesen. Dabei wurde bei jeder Zuweisung das Clusterzentrum neu berechnet. Dieser Vorgang wurde so oft wiederholt, bis sich die Gruppenkonfiguration nicht mehr verändert hat.

Das dezentrale Verfahren beruht auf den gleichen Erkenntnissen. An Stelle des *k-means*-Algorithmus wird die Tatsache berücksichtigt, dass alle Angaben in einem Profil keine konkreten, sondern eher vage Werte sind. Die Basis hierfür ist der in Kapitel 2.6.2 vorgestellten *Fuzzy C-means*-Algorithmus.

Jedoch benötigt auch dieses Verfahren die zufällige Auswahl von Clusterzentren. Diese Auswahl wird nur implizit durch die Agenten getroffen und die Anzahl der Zentren wird zentral bestimmt. Somit kommt es zu einem kleinen Bruch mit der dezentralen Natur des Ansatzes.

Die beiden oben genannten Verfahren *k-means* und *Fuzzy C-means* setzen voraus, dass die Anzahl der zu bildenden Cluster zu Beginn der Analyse feststeht. Das ist wider den dynamischen Charakter des Verfahrens in dieser Arbeit. Es gibt jedoch eine ähnliche Initialisierung. Folgende Dinge werden vorausgesetzt:

1. Die Mindestgröße $|C|_{min}$ eines Cluster C ist global bekannt.
2. Die Maximalgröße $|C|_{max}$ eines Cluster C ist global bekannt.
3. Jeder Agent kennt die Mitglieder seines Clusters.
4. Jeder Agent kennt das Zentrum seines Clusters.

Man sollte sich nun noch einmal das Beispielszenario aus Kapitel 1.4 in Erinnerung rufen. Was passiert eigentlich genau im Kontext der Einladung zu einem gemeinsamen Essen? Jeder Gast erhält eine Einladung. Diese kann angenommen oder abgelehnt werden. Im Fall der Annahme bekommt der Gast eine Bestätigung der Annahme. Ein Server ist demnach auch in diesem Szenario enthalten, jedoch sind seine Dienste eher zweitrangig. Nur die Tatsache, das der Server die Bestätigung verschickt, wird wichtig für die Analyse, denn mit Hilfe dieses Vorgangs werden die vorläufigen Clusterzentren bestimmt. Hierzu wird als erstes bestimmt, wie viele Cluster es maximal geben kann. Die Berechnung erfolgt zentral und ist relativ einfach. Nehmen wir an, es gibt in dem Ferienclub n Gäste. Dann ergibt sich die maximale Anzahl an Clustern C_{max} aus

$$C_{max} = \frac{n}{|C|_{min}}$$

Die zufällige Bestimmung der Clusterzentren erfolgt implizit durch die Agenten. Das Verfahren funktioniert wir folgt:

1. Die ersten C_{max} Agenten, deren inzidente Benutzer die Einladung annehmen, werden zu einem Clusterzentrum.
2. Alle weiteren Agenten erhalten bei Annahme der Einladung die Information, dass es bereits Clusterzentren gibt.
3. All jene Agenten, die nicht Clusterzentrum sind, suchen diese Zentren auf. Dort bewerten sie ihren Grad der Zugehörigkeit zu dem entsprechendem Zentrum.
4. Jeder Agent, der nicht Clusterzentrum ist, ordnet sich dem Zentrum mit dem höchsten Zugehörigkeitsgrad ihrerseits zu.
5. Jedes Mal, wenn ein Agent einem Cluster beitritt, wird das Zentrum neu durch die Agenten bestimmt.
6. Alle einem Cluster zugehörigen Agenten bewerten ihre Zugehörigkeit zu diesem Cluster neu. Gegebenenfalls verlässt ein Agent die Gruppe bzw. teilt sich diese.
7. Solange sich die Gruppenzentren ändern, gehe zu Punkt 3. Ansonsten Abbruch.

Dies wird quasi bis zum Zeitpunkt des Essens so weitergehen. Jedoch kann man feststellen, dass es nach einer gewissen Zeit keine große Aktivität bezüglich der Clusterbildung gibt. Dieser Zustand stellt das Ergebnis der Analyse dar.

Doch wie wird eigentlich bestimmt, in welche Gruppe ein Agent gehört? Alles, was man zur Beantwortung dieser Frage wissen muss, wurde in Kapitel 2.6 erklärt. Im Grunde gibt es in Hinblick auf die Netzwerkaktivität ab dem Zeitpunkt der ersten Teilnahmebestätigung einen starken Aktivitätsanstieg zu verzeichnen. Immer, wenn ein Benutzer die Teilnahme am gemeinschaftlichen Essen bestätigt, tritt sein Agent in Verhandlung mit allen im System aktiven Agenten, deren Benutzer ebenfalls die Teilnahmen bestätigt haben. Bei dieser Kontaktaufnahme vergleichen die Agenten die Profildaten ihrer inzidenten Benutzer. Dabei wird geprüft, zu welchem Grad ein Agent zu dem Cluster des anderen Agenten gehört. Dabei wird folgendes berücksichtigt:

- Ist der Grad der Clusterzugehörigkeit eines Agenten zu einem Cluster c_n größer als der Grad der Clusterzugehörigkeit des Agenten zu einem Cluster c_{n-1}, so wechselt der Agent in den Cluster c_n. Nun überprüft der Agent den Grad der Übereinstimmung seines Profils mit den der anderen dem Cluster zugehörigen Agenten und ermittelt so seine Nachbarn.

- Es findet ein Clusterwechsel eines Agenten aus Cluster c_{n-1} in den Cluster c_n statt, der zur Folge hat, dass die minimale Größe eines Clusters unterschritten wird, so für einen Ausgleich aus dem Cluster c_n gesorgt werden. Dieser findet in der Form statt, dass der Agent mit dem höchsten Clusterzugehörigkeitsgrad zu Cluster c_{n-1} in den Cluster c_{n-1} wechselt.
- Wenn jedoch die Situation kommt, dass der höchste Clusterzugehörigkeitsgrad eines Agenten auf mehrere Cluster zutrifft, dann bleibt der Agent in dem ersten dieser Cluster, in dem Verhandlungen stattgefunden haben.

3.2 Aspekte der Clusterbildung

Hält man sich an dieser Stelle das Beispielszenario aus Kapitel 1.4 und Clusteranalyse im allgemeinen vor Augen. Das Beispielszenario handelt ja von einem Gast, der vom Ferienclub zu einem gemeinsamen Essen mit anderen Gästen des Clubs eingeladen wird. Der Ferienclub hat dabei die Intention, jedem Gast einen möglichst angenehmen Abend zu machen. Wenn man nun die Clusteranalyse mit einbezieht, geht es im Grunde darum, aus allen Gästen Gruppen zu bilden, deren Mitglieder möglichst harmonieren. Dabei gilt es jedoch bestimmte Dinge zu berücksichtigen, die sowohl die Infrastruktur als auch das Verfahren betreffen.

3.2.1 Tischform

Der erste Punkt, den es zu berücksichtigen gilt, ist die Form des Tisches. Es gibt hier beliebig viele Variationen, die eine Gruppenkonfiguration entsprechend erschweren. So kann es beispielsweise sein, dass alle Personen in einer Reihe positioniert werden, was sehr einfach ist.

Eine kompliziertere Form ist ein Tisch, bei dem sich Personen direkt gegenüber sitzen und sich Personen in mehr oder weniger unmittelbarer Hörweite zueinander befinden. Hier muss darauf geachtet werden, dass die in Hörweite zueinander befindlichen Personen ebenfalls harmonieren.

Der Einfachheit halber wurde eine Tischform gewählt, bei der es nur linke und rechte Nachbarn gibt. Der Tisch selber hat die Form eines Kreises. Personen, die gegenüber sitzen, befinden sich außer Hörweite (siehe Abbildung 3.2).

Es gibt noch weitere Sitzordnungen, die denkbar sind. Diese können, wie oben angedeutet, mehr oder weniger starke Komplikationen implizieren. In dieser Arbeit werden jedoch nur

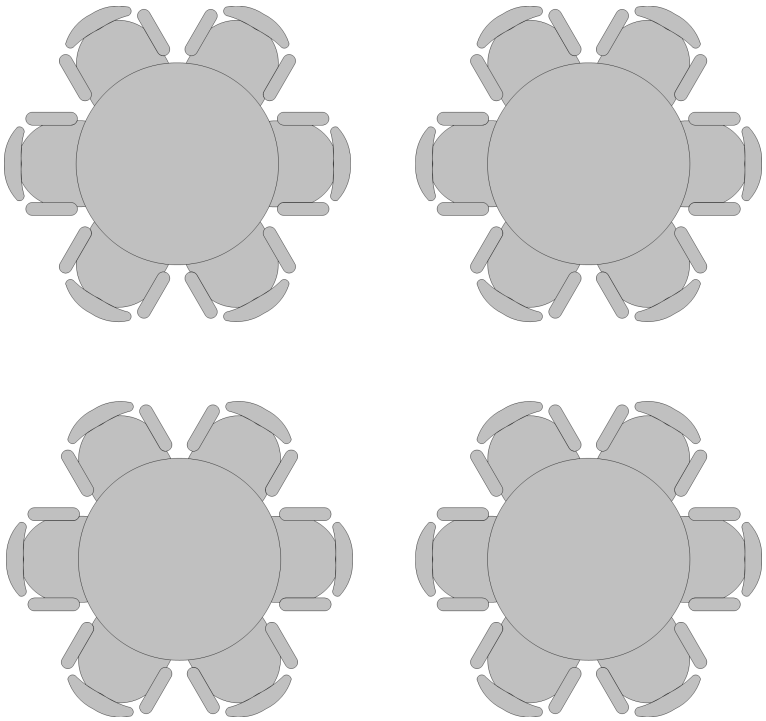

Abbildung 3.2: Ein 4×6-Cluster

Anordnungen der obigen Art getroffen. Von daher werden andere Beispiele an dieser Stelle nicht betrachtet.

3.2.2 Clustergröße

Der zweite wichtige Punkt ist die Größe eines Clusters. Der Einfachheit halber könnte man eine bestimmte Größe für jede Gruppe festlegen. Das impliziert jedoch einen großen Nachteil: Eine feste Clustergröße kann dazu führen, dass Personen zusammen in einer Gruppe sind, deren Profile keinerlei Ähnlichkeit aufweisen, nur um die feste Clustergröße zu erhalten.

Ergo kann die Clustergröße von Gruppe zu Gruppe variieren. Man spricht hier von quasidynamischer Clustergröße. Diese impliziert jedoch mehrere Fragen:

- Wie groß ist ein Cluster mindestens?
- Wie groß ist ein Cluster maximal?
- Wer bestimmt die Größe eines Clusters?
- etc.

In Bezug auf diese Fragestellungen sollte man sich erst einmal über minimale und maximale Größe eines Clusters klar werden. Man spricht in diesem Zusammenhang auch von unterer und oberer Grenze. Eine (sinnvolle) untere Grenze ist für die Clusterbildung gerade in Anbetracht der Forderung an Soziale Software (siehe Kapitel 1.1.3) angebracht. Würde diese Schranke fehlen, gäbe es im schlechtesten Fall genau so viele Gruppen wie Gäste, die jeweils eine Gruppenstärke von einer Person hätten.

In Bezug auf die obere Grenze sind verschiedene Varianten möglich. Betrachten wir als erstes den Fall, dass die Größe eines Clusters nach oben hin nicht beschränkt ist, eine untere Grenze jedoch Gültigkeit hat. Dies hat den Vorteil, dass wirklich alle Personen eines Clusters zusammen an einem Tisch sitzen. Gehen wir zusätzlich davon aus, dass eine obere Grenze die Nachteile einer festen Clustergröße impliziert.

Beispiel Wir haben eine Gruppe von fünf Personen. Es sollen Cluster gebildet werden, für die gilt Clustergröße $x \leq 3$. Ergo werden zwei Cluster gebildet: C_1 und C_2. Nehmen wir weiterhin an, dass wir es mit den aus Kapitel 2.6 bekannten Personen haben und sich folgende Clusterzugehörigkeiten ergeben haben:

Person	**Clusterzugehörigkeit**	
	C_1	C_2
Finja	0,7	0,6
Lilly	0,7	0,7
Mia	0,6	0,1
Paul	0,6	0,1
Tom	0,3	0,8
Louis	0,2	0,9

Tabelle 3.1: Clusterzugehörigkeiten

Bei einer vorgegebenen oberen Grenze würden die folgenden Cluster gebildet werden:

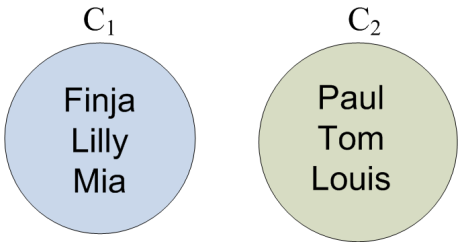

Abbildung 3.3: Clusterbildung mit oberer Grenze

Man erkennt deutlich, dass die Einführung einer oberen Grenze zu suboptimalen Clusterbildungen führt. Paul wurde in diesem Beispiel Cluster C_2 zugewiesen, obwohl dieser nur einen Zugehörigkeitsgrad von $0,1$ hatte. Die Zuweisung erfolgte nur deshalb, da Cluster C_1 schon „voll“ war.

Doch auch das Fehlen einer oberen Grenze hat einen Nachteil. Es kann im schlechtesten Fall dazu kommen, dass mehr Personen an einem Tisch sitzen als überhaupt möglich. Die Annahme, dass eine obere Grenze die Nachteile einer festen Clustergröße impliziert, trifft nicht immer zu. Unter einer Voraussetzung kann und sollte man eine obere Grenze zulassen. Das ist jedoch nur dann der Fall, wenn in Bezug auf die Größe $|C|$ eines Cluster C gilt:

$$|C|_{min} << |C|_{max}$$

Wenn also die minimale Größe eines Clusters signifikant kleiner als die maximale Größe eines Clusters ist, führt das Einfügen einer oberen Grenze dazu, dass, bei einer entsprechenden Anzahl an Teilnehmern, sich ein Cluster teilt, so bald diese oberen Grenze erreicht worden ist.

Beispiel Wir haben eine obere Grenze von acht Personen und eine untere Grenze von drei Personen pro Tisch. Wenn nun ein Cluster "übervoll“ wird, so teilt sich dieser Cluster in zwei Subcluster. Bei neun Personen in einem Cluster hätte man also beispielsweise zwei Tische mit je vier Personen.

3.3 Architektur

Wenn man hinsichtlich einer Systemarchitektur Überlegungen anstellt, sollte man sich erst einmal die lokal involvierten Komponenten des Systems und deren Arbeitsweise in Erinnerung rufen (siehe Kapitel 1 und Kapitel 2).

3.3.1 Arbeitsablauf

In Kapitel 2.5.2 sind mehrere Anwendungsfälle im Kontext des Benutzers vorgestellt worden, die in drei Bereiche unterteilt werden können.

1. Profildaten anlegen, bearbeiten und einsehen.
2. Einladung lesen & Teilnahme annehmen bzw. ablehnen.

3. Reservierung einsehen.

Die grafische Benutzerschnittstelle zu den oben genannten Bereiche stellt das Personal Information Management dar.

Man kann diese Bereiche nochmals unterteilen in lokale und globale Aktivitäten (siehe Tabelle 3.2). Dabei sind globale Aktivitäten jene, die eine Korrespondenz im Clubnetzwerk nach sich ziehen.

Aktivität	
Lokal	**Global**
Profildaten anlegen	Profildaten bearbeiten
Profildaten einsehen	Teilnahme annehmen
Reservierung einsehen	Teilnahme ablehnen

Tabelle 3.2: Lokale und nichtlokale Aktivitäten

Man kann diese Aktivitäten gut in einem Sequenzdiagramm darstellen (siehe Abbildung 3.4), da hier klar ersichtlich ist, was eigentlich passiert.

Profildaten anlegen, bearbeiten und einsehen

Der Benutzer ist mittels der Personal Information Management (PIM) in der Lage, sein Profil zu verwalten. Dazu gehört das Anlegen, Bearbeiten und Einsehen des selbigen. Einsicht und Anlage haben keinerlei Auswirkung auf die übrigen Komponenten des Clubnetzwerkes, insbesondere der anderen Gastendgeräte. Somit handelt es sich um eine lokale Aktivität. Das PIM fordert von dem Profil die Daten des Benutzers an und visualisiert diese. Wenn noch kein Profil existiert, wird ein neues angelegt und dem PIM zur Bearbeitung übergeben.

Wenn das Profil über das PIM bearbeitet werden soll, kann durch die Bearbeitung eine Aktivität ausgelöst werden, die andere Komponenten des Clubnetzwerkes direkt betrifft und fordert. Diese Aktivität ist nicht lokal. Das Auslösen der Aktivität ist jedoch nicht zwingend. Im Grunde öffnet der Benutzer über das PIM sein Profil zur Bearbeitung. Hat er dies gemäß seinen Wünschen modifiziert, sichert er dies. Somit ist noch kein Grund gegeben, warum andere Benutzer beziehungsweise deren Agenten einbezogen werden sollten. Es kann aber sein, dass der Benutzer die Teilnahme am gemeinschaftlichen Essen angenommen hat. In diesem Fall kann es sein, dass - durch Veränderung des Profil - sich Zugehörigkeit zu einem Cluster ändert. Damit verändert sich auch die Sitzordnung. Die Aushandlung einer neuen Sitzordnung bedarf der Verhandlung der benutzerinzidenten Agenten.

Teilnahme annehmen, ablehnen und Reservierung einsehen

In dem Beispielszenario aus Kapitel 1.4 wurde erwähnt, dass der Club seine Gäste zu einem gemeinsamen Essen einlädt. Hierzu wird eine elektronische Einladung an die Endgeräte

der Gäste versandt. Visualisiert wird sie im PIM. Jeder Gast hat nun die Möglichkeit, diese Einladung entweder anzunehmen oder abzulehnen. Lehnt der Gast ab, wird die Ablehnung dem Club mitgeteilt. Dieser bestätigt die Absage. Sollte zum Zeitpunkt der Absage seitens des Gastes schon eine Zusage bestanden haben, so informiert der Agent des Benutzers die möglichen Tischnachbarn, da die Absage zu Folge hat, dass sich die Cluster neu formieren müssen.

Nimmt der Gast die Annahme an, wird die Teilnahme dem Club mitgeteilt und durch diesen bestätigt. Danach beginnt der Agent des Benutzers mit den Verhandlungen, wer Tischnachbar des Benutzers wird. Zu jedem Zeitpunkt hat der Benutzer die Möglichkeit, über das PIM die Reservierung einzusehen und - bis zu einem bestimmten Zeitpunkt - diese zu stornieren.

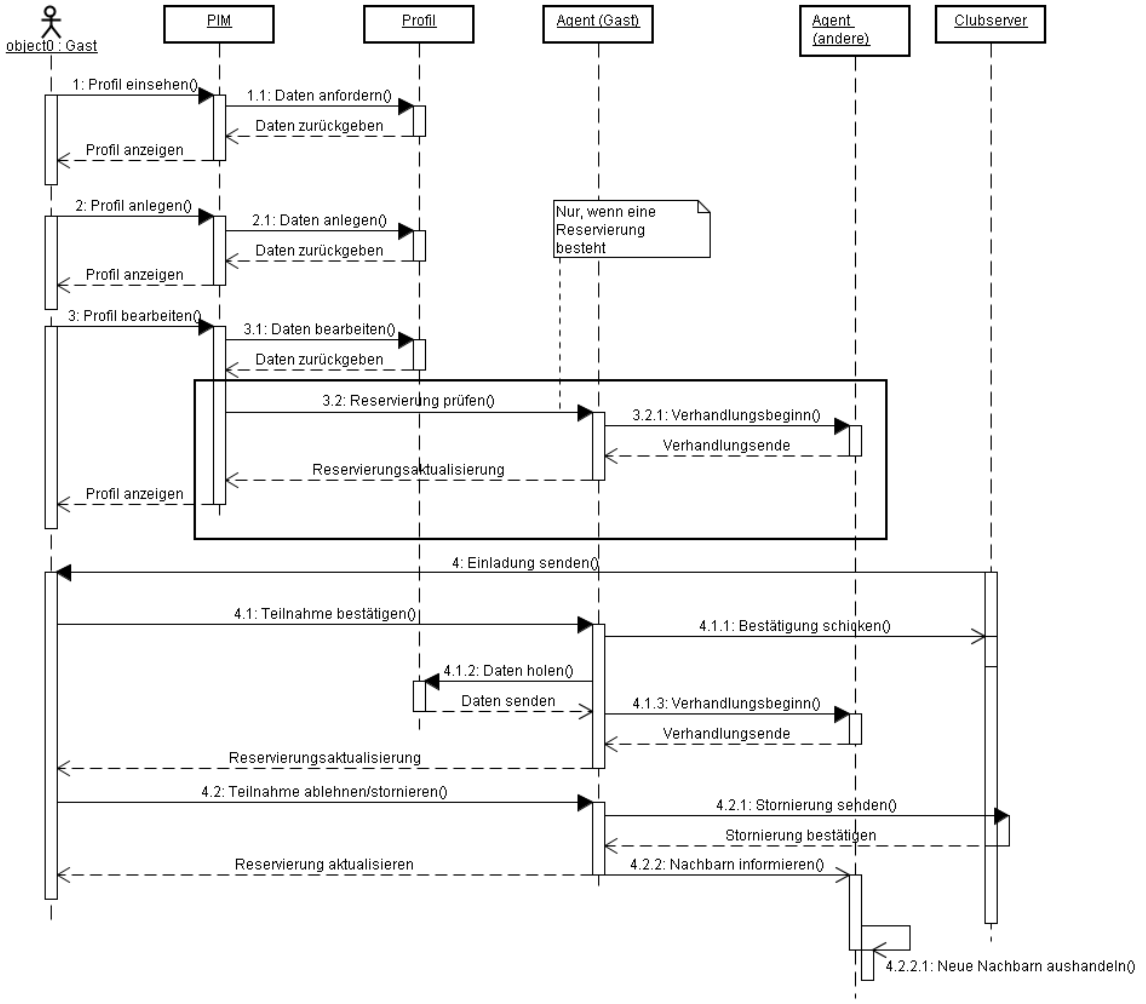

Abbildung 3.4: Der Arbeitsablauf

3.3.2 Personal Information Management

Der Begriff des Personal Information Management (PIM) wurde sehr häufig genannt. Es handelt sich dabei um eine hinsichtlich ihrer Funktionalität äußerst umfangreiche Software. Obwohl diese Software nicht zentraler Gegenstand dieser Arbeit ist, möchte ich an dieser Stelle etwas zum Design des PIM sagen.

Wenn man den Begriff des Personal Information Management rekapituliert, werden mehrere Dinge klar:

- Die Software dient der Verwaltung persönlicher Daten und Informationen und
- die Software verwaltet Informationen im persönlichen Kontext.

Das bedeutet, dass der Umfang der verwalteten Daten als auch die Art der Visualisierung beziehungsweise der Umfang der Funktionalität individuell anpassbar ist. Was bedeutet es für das Design dieser Software? Man kann das PIM als eine Art Kommunikationsmedium verstehen, wobei des PIM selbst keinerlei Funktionalität hinsichtlich der Verwaltung hat. Diese Funktionalität steckt in diversen Komponenten, auf deren Dienste das PIM zugreift und entsprechende Informationen visualisiert. Es wurde in Kapitel 1.4 ein PIM erwähnt, welches seinen Benutzer über Art und Weise der Teilnahme am gemeinschaftlichen Essen informiert. Es handelt sich dabei in der Regel um eine sehr umfangreiche Software, die - da nicht Gegenstand dieser Arbeit - eher rudimentär realisiert wurde. Der Benutzer erhält de facto nur Auskunft über Ort und Datum des Essens, so wie über den Tisch, an dem er sitzt und wer sein linker beziehungsweise rechter Tischnachbar ist.

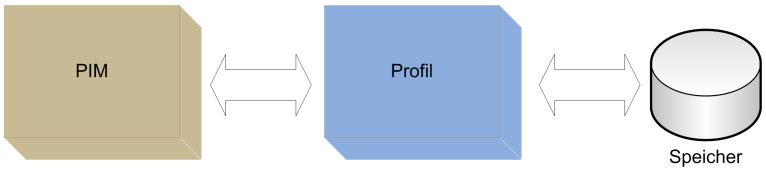

Abbildung 3.5: Informationsfluss zwischen PIM und Profil

Man kann das PIM demnach als grafische Benutzerschnittstelle zu einer frei konfigurierbaren Menge von Anwendungen sehen. Hinsichtlich der Realisierung des PIM kann man auf die Erfahrung anderer zurückgreifen, die für bestimmte Situationen sogenannte Entwurfsmuster vorgestellt haben. Dabei ist ein Entwurfsmuster nicht dogmatisch zu verstehen, sondern lediglich als Hilfestellung oder „Kochrezept“, welches sich in der Praxis bewährt hat. Folgende Dinge können effizient mit Entwurfsmustern realisiert werden:

- Grafische Benutzerschnittstelle und
- freie Konfigurierbarkeit.

Diese Muster sollen an dieser Stelle der Orientierung dienen. Wer mehr über die genannten Entwurfsmuster wissen möchte oder generell etwas über wiederverwendbare Konzepte lernen will, dem sei (Gamma u. a., 1996; Eilebrecht und Starke, 2004) ans Herz gelegt. Diese Werke sind auch die Quelle der folgenden Muster.

3.3.2.1 Model-View-Controller

Dieser Ansatz wird zur Erstellung von Benutzerschnittstellen verwendet. Dabei geht es darum, die einzelnen Komponenten sauber zu kapseln und Verantwortlichkeiten dort zu belassen, wo sind semantisch beziehungsweise syntaktisch auch hingehören. Man hat es hier mit den Klassen

- Model,
- View und
- Controller zu tun.

Das Model-Objekt stellt in diesem Kontext das Anwendungsobjekt dar, das View-Objekt stellt eine Visualisierungsvariante dar und das Controller-Objekt bestimmt die Möglichkeiten, mit denen das View-Objekt auf das Model-Objekt zugreifen kann. Mit dem Model-View-Controller-Paradigma (MVC) werden die einzelnen Komponenten entkoppelt (siehe Abbildung 3.6), was zu einer Steigerung von Flexibilität und Wiederverwendbarkeit führt. So ist es denkbar, ein Model-Objekt mit verschiedenen View-Objekten zu visualisieren, zum Beispiel mittels Java-Swing und Java Server Faces (JSF).

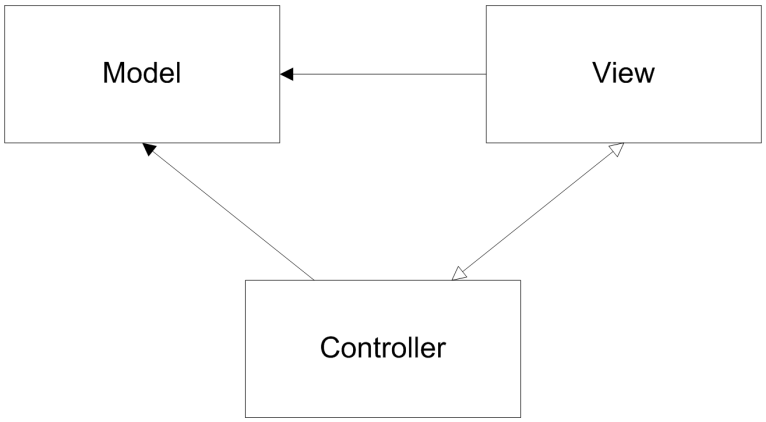

Abbildung 3.6: Model-View-Controller-Architektur

Dieses Muster hat jedoch auch Nachteile. Der Implementierungsaufwand erhöht sich, denn die Unabhängigkeit der einzelnen Komponenten sorgt gleichzeitig dafür, dass gleichzeitige

Anzeige und Manipulation eines Model-Objektes in unterschiedlichen View-Objekten zu einer Inkonsistenz der Daten führen kann. Dieses Problem ist erkannt und gelöst worden.

3.3.2.2 Beobachtermuster

Die Gefahr der Dateninkonsistenz des MVC-Musters kann mit dem sogenannten Beobachtermuster gelöst werden. Im Grunde muss beim MVC-Muster das View-Objekt sicherstellen, dass seine Darstellung dem Zustand des Model-Objektes entspricht. Dazu benachrichtigt das Model-Objekt alle von ihm abhängigen View-Objekte, wenn sich seine Daten ändern. Somit wird jedes View-Objekt in die Lage versetzt, sich selbst in einen konsistenten Zustand zu versetzen.

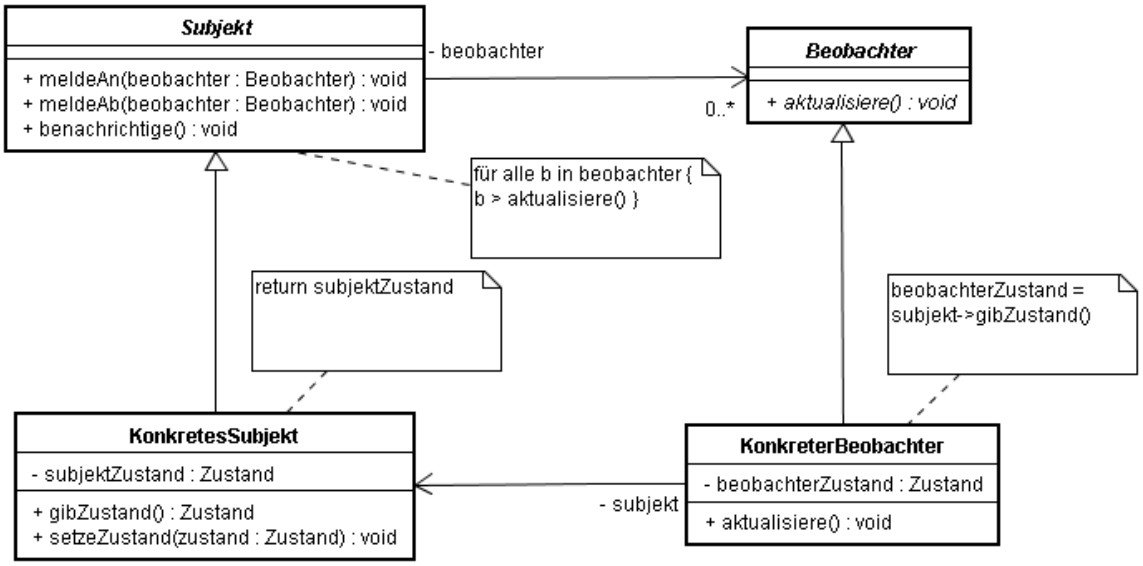

Abbildung 3.7: Das Beobachtermuster

Das MVC- und das Beobachter-Muster stellt das Design der Benutzerschnittstelle dar. Jedoch muss das Problem der freien Konfigurierbarkeit gelöst werden.

3.3.2.3 PlugIn-Muster

An dieser Stelle muss man sich erst einmal darüber im Klaren sein, was freie Konfigurierbarkeit eigentlich heißt. In dieser Arbeit wird es dahingehend interpretiert, dass man das PIM um beliebige Komponenten einfach erweitern kann. Die Zusammenarbeit der Komponenten mit dem PIM als auch untereinander ist über Schnittstellen definiert. Damit wird man auch

einem Teil der allgemeinen Anforderungen aus der Informatik gerecht (siehe Kapitel 2.3). Diese einfach einer Anwendung hinzufügbaren Komponenten nennt man „PlugIns“. Auch hier gibt es ein Muster, welches sich durchaus bewährt hat: das PlugIn-Muster.

Der Vorteil ist, dass PlugIns zur Laufzeit eingebunden werden können und nicht nur durch einen Übersetzungsprozess der Software. Das hat eine Erhöhung der Erweiterbarkeit und der Anpassungsfähigkeit zur Folge. Somit kann die Entwicklung von Erweiterungen unabhängig von der Entwicklung der eigentlichen Applikation stattfinden. Die Arbeitsweise des Musters kann man Abbildung 3.8 entnehmen. Wie das Muster strukturell aufgebaut ist, wird in Abbildung 3.9 deutlich.

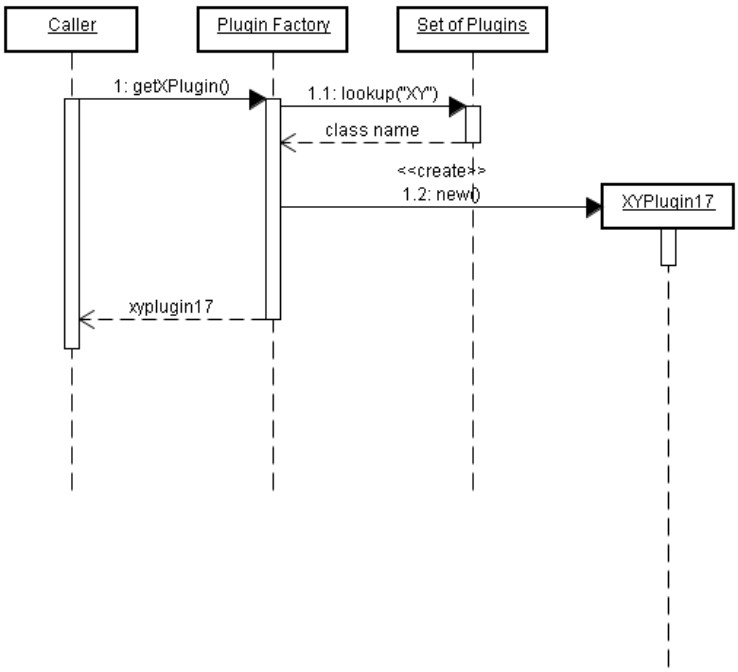

Abbildung 3.8: Arbeitsweise des Plugin-Musters

3.3.3 Das Profil

Mit dem Profil haben wir uns in Kapitel 2.6 eingehend beschäftigt. Dort wurde deutlich, dass in unserem Fall ein Profil identisch mit einem Vektor ganzer nichtnegativer Zahlen ist. Auch die Tatsache, dass jede Zahl die Ausprägung einer Eigenschaft in Bezug auf die eigene Person ist, hat man an dieser Stelle erfahren. Es handelt sich also bei den Elementen des Vektors quasi um Wert-Schlüssel-Paare, wobei der Wert die Ausprägung einer Eigenschaft und der Schlüssel dieser Eigenschaft ist. Da die Eigenschaft an sich eher natürlichsprachlich formuliert wird und nicht unbedingt auf eine natürliche nichtnegatve ganze Zahl abgebildet

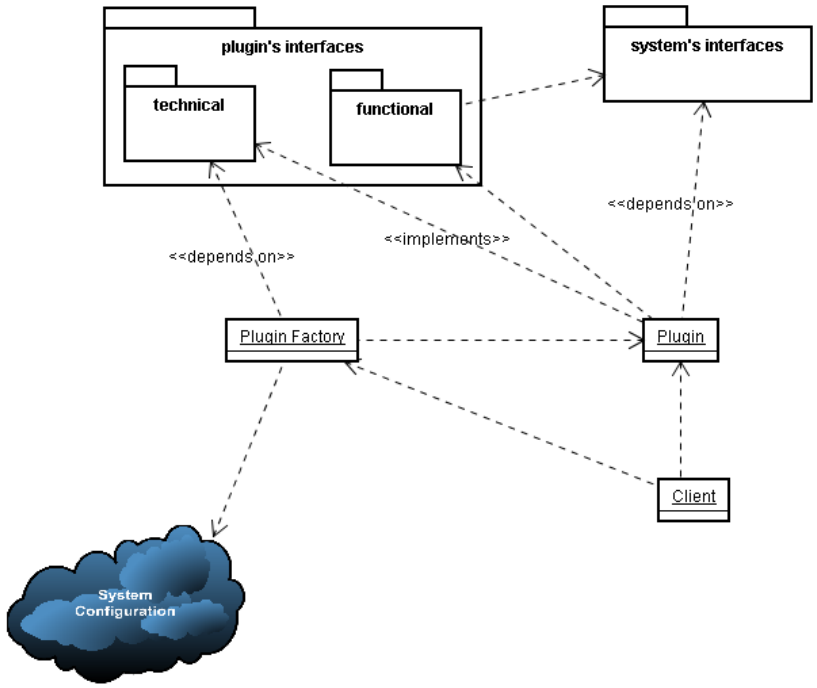

Abbildung 3.9: Struktur des Plugin-Musters

wird, eignet sich zur Präsentation auf dem Computer eine Datenstruktur mit frei definierbaren eindeutigen Schlüsseln und frei formulierbaren Werten. Diese Datenstruktur ist unter dem Begriff des Dictionary[1] bekannt.

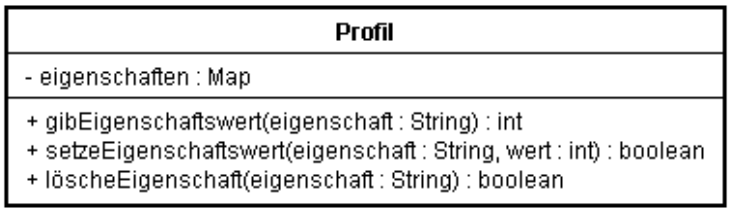

Abbildung 3.10: Klassendiagramm Profil

Da man Profile dahingehend erweitern könnte, dass ein Profil kategorisierbar ist, also Unterprofile enthält, ist der Einsatz eines Entwurfsmusters naheliegend, welche diese Teil-Ganzes-Beziehung unterstützt: das Kompositum.

Dieses Entwurfsmuster fügt Objekte zu Baumstrukturen zusammen, um diese Beziehungen hierarchisch darzustellen. Es ermöglicht Klienten die einheitliche Behandlung von Objekten

[1] In Java heißt die entsprechende Datenstruktur *Map*.

als auch Kompositionen von Objekten. Eine typische Objektstruktur des Kompositums stellt Abbildung 3.11 dar.

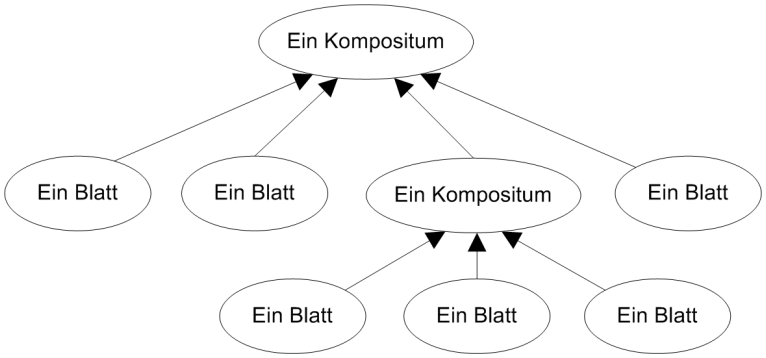

Abbildung 3.11: Typische Objektstruktur des Kompositums

Man sieht in Abbildung 3.11 sehr deutlich, was unter hierarchischer Beziehung und der Baumstruktur zu verstehen ist: Eine Instanz einer Klasse besteht aus mehreren Teilen. Dabei kann ein Einzelteil wiederum eine Instanz derselben Klasse sein. Die grundlegende Struktur wird in Abbildung 3.12 deutlich.

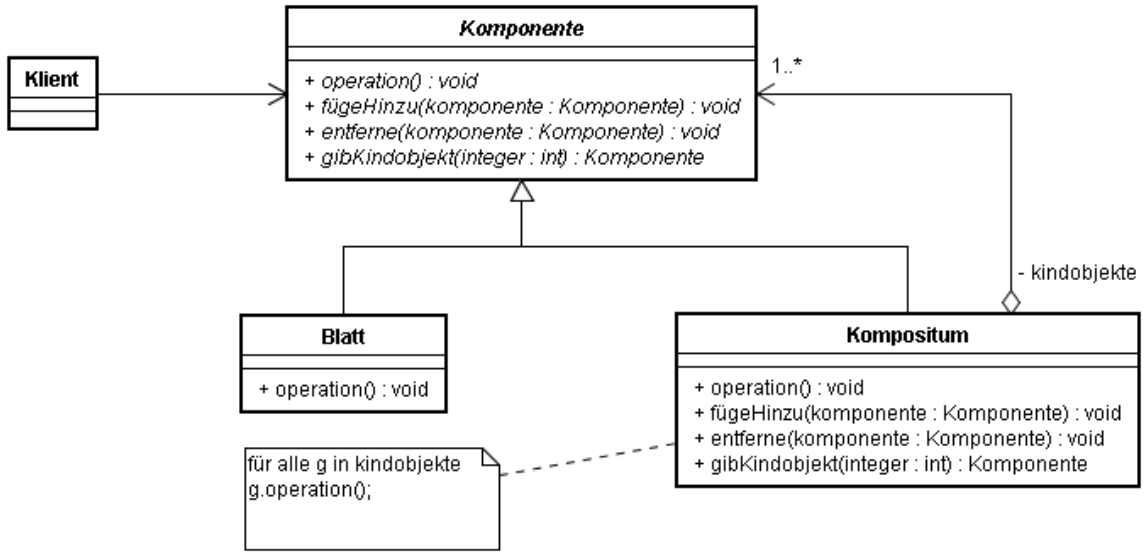

Abbildung 3.12: Grundlegende Struktur des Kompositums

Somit ist die Frage der Realisierung des Profils geklärt. Das Profil dient jedoch nicht der Verwaltung von Benutzerdaten. Ein entsprechendes Objekt, welches den Benutzer repräsentiert, muss zwar implementiert werden, da auf das Profil darüber letztendlich zugegriffen wird, jedoch ist der Umfang der zu verwaltenden Daten beliebig und im Grund für diese Arbeit nicht von Bedeutung. Von daher ist dieser Aspekt vernachlässigbar.

3.3.4 Der Softwareagent

Es gibt jedoch noch einige Komponenten, denen man eine gewisse Aufmerksamkeit widmen muss. Eine davon ist die Komponente, die letztendlich die eigentliche Clusterkonfiguration aushandelt: der Softwareagent.

Die Basis zur Konstruktion des Softwareagenten stellt das *Construction Intelligent Agents*-Framework (*CIAgent*) von (Bigus und Bigus, 2001) dar. Diese Framework genügte zwar den meisten Anforderungen, musste im Detail jedoch mehr oder weniger stark angepasst werden. Der Softwareagent besteht aus mehreren Klassen, welche in einem Paket *Clusteragent* zusammengefasst sind (siehe Abbildung 3.13).

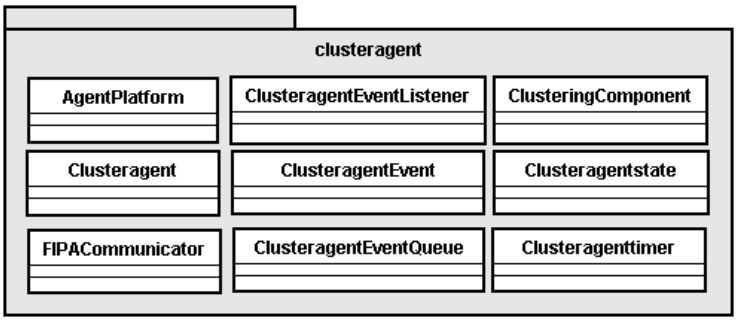

Abbildung 3.13: Paketstruktur des Softwareagenten

Wie diese einzelnen Klassen miteinander in Beziehung stehen, soll das nachfolgende Klassendiagramm zeigen (siehe Abbildung 3.14).

Zentrale Klasse in dem Framework ist die Klasse **Clusteragent** (siehe Abbildung 3.15). Sie legt die gemeinsame Programmierschnittstelle und das Verhalten der mit dem Framework entwickelten Softwareagenten fest. Abbildung 3.14 kann man auch entnehmen, welche Klassen eine funktionierende Clusteragentinstanz ausmachen.

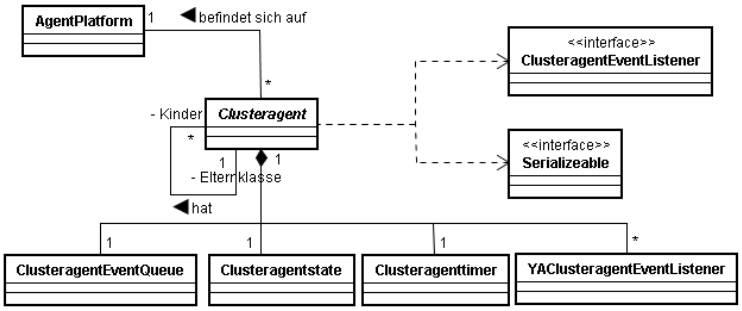

Abbildung 3.14: Klassendiagramm des Softwareagenten

Neben der Klasse Clusteragent gibt es noch weitere existentielle Klassen in dem Framework, deren Funktionsweisen und Bedeutungen an dieser Stelle kurz erläutert werden sollen.

Die Methoden dieser Klasse sind in den meisten Fällen eine Delegation an die Bestandteile eines Agenten. Die wichtigsten Methoden werden in den einzelnen Klassen an sich kurz vorgestellt.

Clusteragentstate

Die Klasse **Clusteragentstate** repräsentiert den Zustand eines Agenten. Im Moment der Konstruktion ist sein Zustand *UNINITIATED*. Im Lauf des Lebenszyklus eines Agenten kann sich dieser jedoch seinen Zustand ändern. Es gibt die Zustände

1. UNINITIATED,
2. INITIATED,
3. ACTIVE,
4. SUSPENDED und
5. UNKNOWN.

Die Methoden, die der Klasse **Clusteragentstate** zur Verfügung stehen, können dem zugehörigen Klassendiagramm entnommen werden (siehe Abbildung 3.16).

Methode	**Beschreibung**
Clusteragentstate()	Konstruktor einer Instanz von **Clusteragentstate**
setState(int state)	Setzt den Zustand eines Agenten.
getState()	Gibt die Ausprägung des Zustands zurück.
toString()	Überführt die Ausprägung des Zustandes von einem Zahlenwert in einen String.

Tabelle 3.3: Methodenübersicht der Klasse Clusteragentstate

Clusteragenttimer

Die geforderte Autonomie eines Agenten wird in dieser Klasse realisiert. Ebenfalls erhält ein Agent durch diese Klasse die Fähigkeit zur asynchronen Ereignisverarbeitung. Autonomie erhält der Agent durch Implementierung **Clusteragenttimer** der *Runnable*-Schnittstelle. Asynchrone Ereignisverarbeitung wird dadurch erreicht, dass Ereignisse in eine Warteschlange gestellt werden, welche dann in bestimmten Zeitintervallen verarbeitet werden.
Beide Verhaltensweisen werden von einem einzigen Thread unterstützt.

Neben diesen Methoden ist noch der Konstruktor *Clusteragenttimer(...)* enthalten.

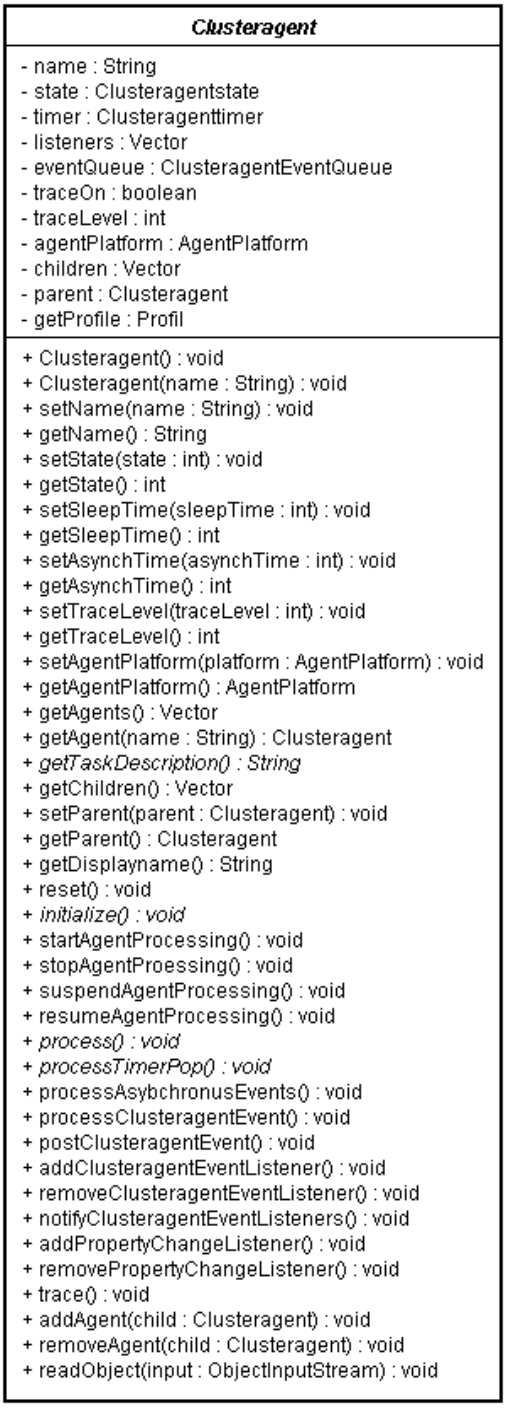

Abbildung 3.15: Die Klasse Clusteragent

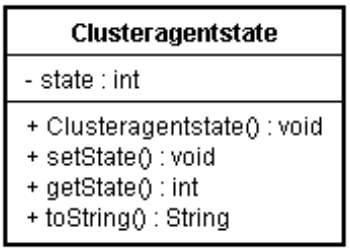

Abbildung 3.16: Die Klasse Clusteragentstate

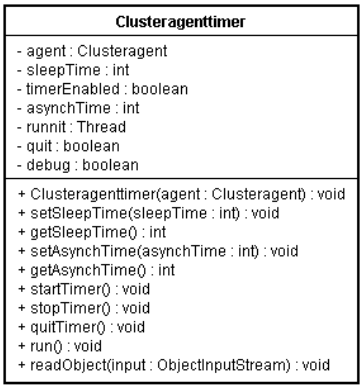

Abbildung 3.17: Die Klasse Clusteragenttimer

Methode	**Beschreibung**
run()	Da es sich hier um einen nebenläufigen Prozess (Thread) handelt, wird die Schnittstelle *Runnable* implementiert.
processTimerPop()	Das autonome Verhalten. Der Agent wird durch diese Methode jede *sleepTime*-Millisekunde aufgerufen-
startTimer()	Startet den Timer zur asynchronen Ereignisbearbeitung alle *asyncTime*-Millisekunden.
stopTimer()	setzt *timerEnabled* auf *false*.
quitTimer()	Setzt *quit* auf *true*. Das führt zur Beendigung von *run()*.
setSleepTime(int sleepTime)	Setzt den Wert von *sleepTime*.
getSleepTime()	Gibt den Wert von *sleepTime* zurück.
setAsyncTime(int asyncTime)	Setzt den Wert von *asyncTime*.
getAsyncTime()	Gibt den Wert von *asyncTime* zurück.

Tabelle 3.4: Methodenübersicht der Klasse Clusteragenttimer

ClusteragentEventListener

Diese Klasse dient der Kommunikation mit anderen Agenten. Hierzu wird die Schnittstelle *EventListener* implementiert. Die Klasse **Clusteragent** enthält einen *Vektor* von *Listenern*. Bei den *Listenern* handelt es sich um alle Klassen, die die Schnittstelle **ClusteragentEventListener** implementieren. Dies ermöglicht, dass quasi jedes Java-Objekt Quelle eines Ereignisses sein kann.

Abbildung 3.18: Die Klasse ClusteragentEventListener

Methode	Beschreibung
postClusteragentEvent(...)	Legt ein Ereignis in der Warteschlange abzuarbeitender Ereignisse ab.
processClusteragentEvent(...)	Führt zur sofortigen Verarbeitung eines Ereignisses innerhalb des aufrufenden Threads.

Tabelle 3.5: Methodenübersicht der Klasse ClusteragentEventListener

ClusteragentEvent

Es handelt sich hierbei um die Repräsentation eines Ereignisses. Wie diese Klasse aufgebaut ist, kann man in Abbildung 3.19 sehen.

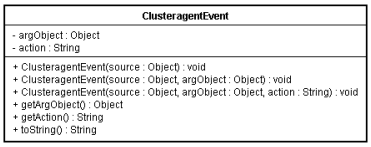

Abbildung 3.19: Die Klasse ClusteragentEvent

Die Methoden dieser Klasse werden durch den jeweiligen Namen weitestgehend erklärt, weshalb eine nähere Erläuterung an dieser Stelle nicht notwendig ist.

ClusteragentEventQueue

ClusteragentEventQueue realisiert eine Warteschlange von **ClusteragentEvents**. Der Aufbau dieser Klasse ist relativ einfach, wie man in Abbildung 3.20 sieht.

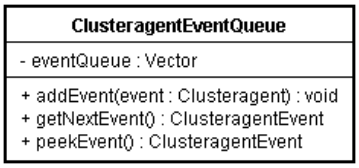

Abbildung 3.20: Die Klasse ClusteragentEventQueue

Methode	Beschreibung
addEvent(...)	Fügt ein Ereignis zur Warteschlange hinzu.
getNextEvent()	Holt das nächste abzuarbeitende Ereignis.
peekEvent()	Holt das erste Element der Warteschlange. Wenn diese leer ist, wird *null* zurückgegeben.

Tabelle 3.6: Methodenübersicht der Klasse ClusteragentEventQueue

3.3.5 Die Clusteringkomponente

Die Klasse **Cluster** (siehe Abbildung 3.21) realisiert die Analyse des Clusters und regelt auch, ob ein Agent linker bzw. rechter Nachbar wird oder nicht.

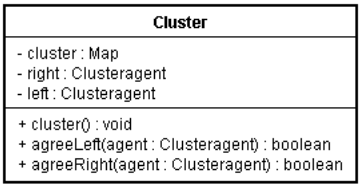

Abbildung 3.21: Die Klasse Cluster

Dabei wurden die Erkenntnisse aus Kapitel 3.1 umgesetzt.

3.3.6 Kommunikation

Es wurde versucht, in diesem Agentensystem eine Kommunikation via *FIPA* zu realisieren. Als Hilfestellung diente hier das *FIPA-OS*-Toolkit. Die Komplexität dieses Tools machte es jedoch nicht möglich, eine Realisierung in der gegebenen Zeit fertigzustellen. Alternativ erfolgte die Kommunikation über *Java-Beans*. Im Falle einer Realisierung sollte *FIPA-OS* jedoch integriert werden (Weiß und Jakob, 2005; Bigus und Bigus, 2001).

3.4 Umsetzung

In diesem Kapitel wird erläutert, mit welchen Mittel die prototypische Realisierung der multiagentensystemgestützten Clusteranalyse erfolgt ist. Auch sollen Probleme der Umsetzung an geeigneter Stelle diskutiert werden.

Wenn man an die Umsetzung des Inhalts der vergangenen Kapitel denkt, vor allem mit Blick auf Kapitel 2, so muss man sich über mehrere Dinge bezüglich der Umsetzung klar werden:

- Welche Rechnerarchitektur verwende ich?
- Auf welchen Plattformen arbeite ich?
- Welche Programmiersprache verwende ich?

Diese Fragen können zum Teil schon im Vorfeld beantwortet werden.

3.4.1 Rechnerarchitekturen

Wie in Kapitel 1.3 oder in der Vision von Butrynowski (2005) erwähnt wurde, ist das Ziel die Umsetzung der Applikation auf möglichst kleine Geräte. Damit kann ubiquitäres oder auch Ambient Computing gerecht werden, da die Geräte so weit im Kontext der Forderung des Benutzers im Hintergrund verschwinden, dass der Umgang mit ihnen so selbstverständlich wie Essen, Schlafen oder ähnliches wird. Dennoch unterstützen diese den Benutzer so gut wie möglich. Um eine Art Evolution zu verzeichnen, wurde anfangs eine quasi normale, alltägliche Rechnerform genutzt: der herkömmliche Desktopcomputer. Schrittweise wurde die Umgebung dann dekrementiert, so dass man über Laptops und Subnotebooks hin zu einem modernen iPaq H5500 PDA der Firma Hewlett Packard entwickelte.

3.4.2 Plattformen

Getreu dem Motto „Einmal schreiben - überall laufen lassen“ wurde in Bezug auf diesen Punkt gleich der Gedanke in Richtung der verwendeten Programmiersprache aufgegriffen. So ist das Ziel, eine Applikation zu schaffen, die möglichst unproblematisch auf so vielen Plattformen wie möglich laufen soll. Da gab es im Grunde nur eine Möglichkeit: Java.

3.4.3 Java

Java ist eine verhältnismäßig junge Programmiersprache, die jedoch trotz des niedriges Alters eine turbulente Vergangenheit hat. Den Grundstein legte eine Gruppe von Ingenieuren bei Sun Microsystems, die im Green-Projekt arbeiteten. Dieses Projekt beschäftigte sich mit der Entwicklung von interaktiven Fernsehen. Neben dem Betriebssystem „Green-OS“ war auch ein Interpreter namens „OAK“ Bestandteil des Projekts. „OAK“ ist der Vorläufer des heutigen Java.

Als dann im Jahre 1994 das World Wide Web aufkam, erkannte man das Potential dieses Mediums und entwickelte einen Webbrowser (WebRunner bzw. HotJava), der die Möglichkeit bereitstellen sollte, Programme aus dem damals jungen WWW direkt zu starten. Um zusätzlich zukünftige Entwickler zu gewinnen, erfuhr „OAK“ eine Metamorphose in Richtung C++, damit diejenigen Entwickler, die bis dato ihre Applikationen in C++ implementierten, nach Java, so der jetzige Name des mutierten „OAK“, zu locken.

Der eigentliche Durchbruch von Java erfolgte jedoch erst im Jahre 1995 mit der Lizensierung von Netscape. Im Jahre 1996 erfolgte die Freigabe des ersten Java Development Kits (JDK) (Böhm, 2002).

Mittlerweile liegt Java in drei verschiedenen Editionen vor:

- Der Java 2 Enterprise Edition J2EE,
- der Java 2 Standard Edition J2SE und
- der Java 2 Micro Edition J2ME.

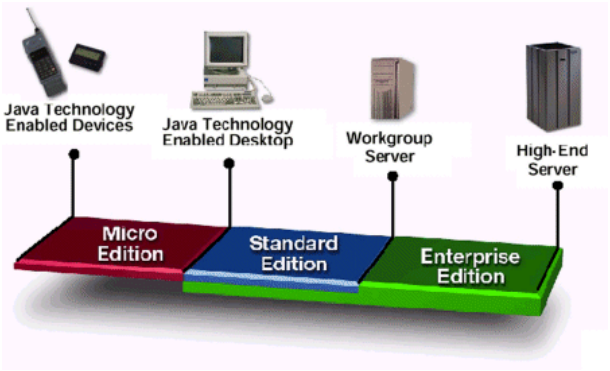

Abbildung 3.22: Die drei Java Editionen im Überblick

3.4.3.1 Java 2 Enterprise Edition

Die Java 2 Enterprise Edition (J2EE) ist ein Framework zur Entwicklung verteilter Anwendungen und erweitert die J2SE um die dafür notwendigen Komponenten. Sie besteht aus 3 Komponenten:

- Client Components,
- Web Components und
- Business Components.

Unter den Clientkomponenten versteht man Applets und Applikationen, die auf der Clientseite ausgeführt werden. Die Komponenten, die serverseitig ausgeführt werden, bezeichnet man unter anderem als Servlets oder Java Server Pages (JSPs). Die Businesskomponenten, die sogenannten Enterprise Java Beans (EJBs) dienen zur Kapselung von Daten. Mit ihnen kann Persistenz und Verteilung von Java-Objekten erreicht werden. J2EE-Applikationen bestehen immer aus einem Server und mindestens einem Client. Der Server ist ein sogenannter Applikationserver, der die J2EE-Architektur realisiert.

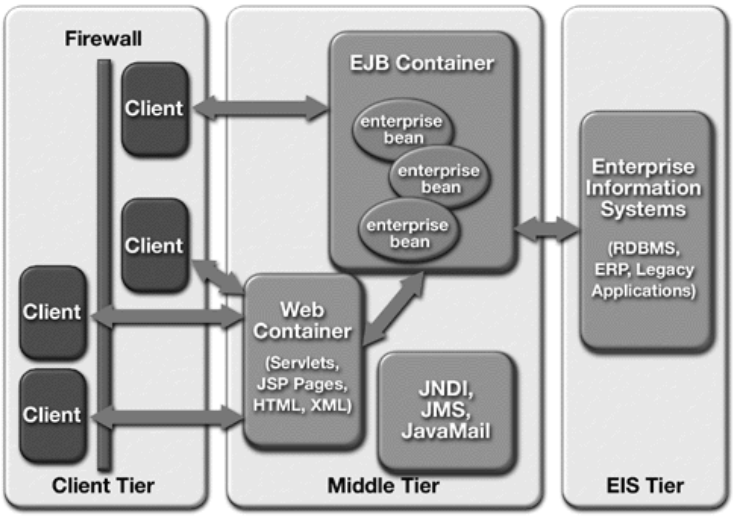

Abbildung 3.23: Java 2 Enterprise Edition

3.4.3.2 Java 2 Standard Edition

Die Java 2 Standard Edition (J2SE) ist die Grundlage der J2EE. In der Standardedition sind das sogenannte Source Development Kit (SDK)[2] und die Java Laufzeitumgebung (JRE). Diese bilden die Grundlage zur Anwendungsentwicklung von Desktop- und Serverapplikationen.

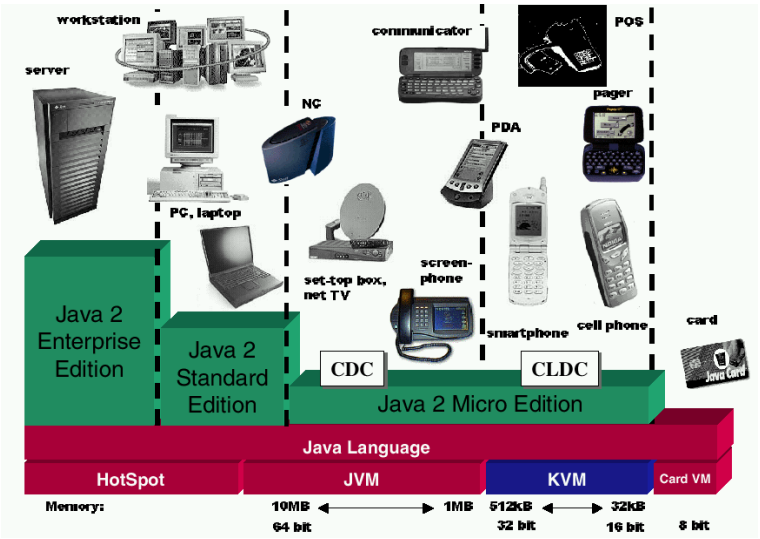

Abbildung 3.24: Die drei Editionen im Kontext

3.4.4 Java 2 Micro Edition

Die Java 2 Micro Edition (J2ME) deckt als Clientplattform den Bereich der clientseitigen Lösungen von Softwarearchitektur ab. Für den Einsatz dieser Java-Edition sprechen mehrere Gründe. Im Bereich der Mobilgeräte ist der Einsatz der J2ME relativ weit verbreitet, so dass hier eine Unterstützung durch alle gängigen Hersteller vorzufinden ist. Auch die Bekannten Aspekte des Einsatzes von Java im Allgemeinen wie Plattformunabhängigkeit und Portabilität sind, auch wenn es sich um eine eingeschränkte Version der Java-Edition handelt, gewährleistet. Auf die Einschränkungen wird in einem späteren Kapitel eingegangen. Zu diesen Vorteilen kommt ein ausgefeiltes Sicherheitskonzept.

In Hinblick auf das erstgenannte Argument spricht noch ein weiterer Faktor für den Einsatz dieser Java-Edition. Laut einer Schätzung der ARC Group sind im Jahre 2002 weltweit 31 Millionen und im Jahre 2003 88 Millionen Java-fähige Mobiltelefone verkauft worden. 2008 werden sogar 592 Millionen verkaufte Java-fähige Mobiltelefone erwartet. Diese Prognose,

[2] Das SDK (Source Development Kit) löst die alte Bezeichung JDK ab.

die in der Einleitung erwähnte Studie und die zunehmnende Symbiose von Mobiltelefon und PDA lassen darauf schließen, dass fast das siebenfache aller Nutzer mobiler Endgeräte in der Lage sein wird, leistungsfähige, individualisierte Softwaresysteme zu nutzen.

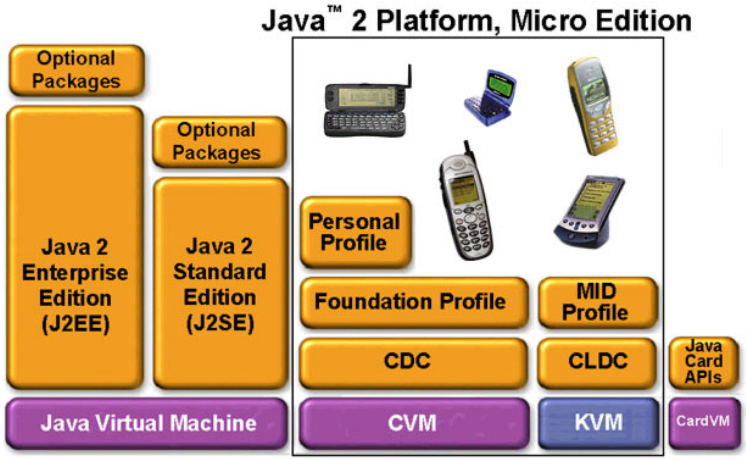

Abbildung 3.25: J2ME im Überblick

Auch das Einsatzspektrum der J2ME spricht für ihren Einsatz, reicht es doch vom Mobiltelefon bis zur Set-Top-Box, einem Gerät zum Empfang digitaler Fernsehkanäle, welche später erweitert wurden, um Fernsehgeräte internetfähig zu machen.

Ebenso sprechen auch betriebswirtschaftliche Gründe für den Einsatz der J2ME. Ein Überschussangebot führt zum Preisverfall der Mobiltelefone. Dies führt dazu, dass Einsparungen in den Herstellungskosten gemacht werden. Diese sind nicht nur personellen Ursprungs. Es führt auch zum Einsatz kostengünstiger Komponenten, was wiederum zu einer Limitierung von Rechenleistung und Speicherkapazität führt. Die Fähigkeit von J2ME, auf in Hinblick auf Resourcen limitierten Geräten zu funktionieren, spricht für ihren Einsatz.

Prinzipiell unterstützt J2ME zwei Gruppen von Geräten:

- Persönliche, vernetzte Geräte
- Gemeinsam genutzte, stationäre, vernetzte Geräte

Der Fokus dieser Arbeit liegt auf den Geräten der ersten Gruppe.

3.4.4.1 Nachteile

Doch der Einsatz der J2ME bringt auch Nachteile mit sich. So decken die in der J2ME spezifizierten Programmierschnittstellen nur einen Teil der Funktionalität eines modernen Endgerätes ab. Herstellerspezifizierte Java-Bibliotheken gleichen dieses Manko zwar (zumindest)

teilweise aus, doch die Nutzung des übrigbleibenden Teils der Funktionalität ist nur über native Programme via Nutzung proprietärer Schnittstellen möglich.

Ein weiterer Nachteil sind Performanzeinbußen gegenüber nativen Programmen, da J2ME-Programme interpretativ ausgeführt werden und ein Just-in-time-Compiler fehlt. Das führt dazu, dass zeitkritische Anwendungen und Komponenten nativ implementiert werden.

3.4.4.2 Vorteile

Ein großer Vorteil ist die eingangs erwähnte weite Verbreitung der J2ME. Dies führt auch dazu, dass sich für diese Plattform entwickelte Applikationen nahezu problemlos auf unterschiedlichen Endgeräten ausführen lassen. Es wird dadurch ein hoher Grad an Portabilität erreicht. Die Spezifikationen der J2ME werden im Java Community Process (JCP) zwischen den Herstellern abgestimmt und sind offen zugänglich und wohl dokumentiert. Dadurch ist es relativ einfach, die J2ME durch (OEM-)spezifische Klassenbibliotheken zu erweitern.

Abstrahiert man die Betrachtungsweise um eine weitere Stufe und richtet den Fokus von Schnittstellen und Implementation auf die Spracheigenschaften, so ist auch hier die Nutzung von Java und das damit einhergehende hohe Abstraktionsniveau, so wie der Verzicht auf fehleranfällige Konstrukte Grund für eine höhere Produktivität und geringere Fehleranfälligkeit. Überdies benötigt der erfahrene Java-Entwickler nur kurze Einarbeitungszeit, um mit der J2ME Applikationen zu entwickeln.

3.4.4.3 Architektur

J2ME besteht im wesentlichen aus drei Komponenten:

- Konfiguration
- Profil(e)
- Optionale Pakete

Dabei stellt die Konfiguration die Basis von J2ME dar, auf der die Profile aufsetzen. Wenn mobile Endgeräte mit diesen beiden Schichten auskommen und diese Gesamtkonfiguration herstellerübergreifend ist, stellt die Portierung von Software auf Engeräte verschiedener Hersteller kein Problem mehr dar. Schwierig wird es jedoch, wenn zu den beiden genannten Schichten ein dritte hinzukommt: die optionalen Pakete.

3.4.4.4 Konfiguration

Die Konfiguration stellt den Kern der J2ME dar. In ihr sind die unterstützten Sprachmittel von Java , der Funktionsumfang der Java Virtual Machine (JVM) und die nutzbaren Klassenbibliotheken deklariert. Es gibt zur Zeit zwei unterschiedliche Konfigurationen:

- Connected, Limited Device Configuration (CLDC) und
- Connected Device Configuration (CDC).

Die Wahl der Konfiguration hängt vom Endgerät ab. Resourcenlimitierte Endgeräte nutzen die CLDC, andere die CDC. Näheres kann in (Schmatz, 2004) nachgelesen werden.

3.4.4.5 Profil(e)

Die Profile erweitern die Konfiguration um leistungsspezifische Merkmale, wie zum Beispiel PIM. Auch hier gibt es zur Zeit zwei Varianten:

- Mobile Information Device Profile (MIDP) und
- Information Module Profile (IMP).

Da diese Profile in diesem Kontext nicht näher erläutert werden müssen, kann an dieser Stelle auf einschlägige Fachliteratur wie zum Beispiel (Schmatz, 2004) verwiesen werden.

3.4.4.6 Optionale Pakete

Die optionalen Pakete gehen über die Erweiterungen der Profile hinaus. Obwohl diese häufig OEM-Klassenbibliotheken beinhalten, stellt dies noch kein Problem dar, da diese Pakete im JCP diskutiert werden. Das eigentliche Problem der optionalen Pakete ist, dass deren Implementierung nicht Pflicht ist. Somit kann die Nutzung dieser Pakete zu Einschränkungen der Portabilität führen.

Beispiele für optionale Pakete sind u.a.

- PDA Optional Packages,
- Java API für Bluetooth oder
- Wireless Messaging API.

3.4.4.7 Schwierigkeiten im Einsatz

Im Vergleich zu J2EE oder J2SE ist der Einsatz der J2ME für Programmierer, auch für solche mit Erfahrung in Java, nicht ganz einfach. Ein großes Problem ist die Mächtigkeit von J2ME bzw. deren Limitierung im Hinblick auf die anderen beiden Editionen. Die Mächtigkeit von J2ME ist in folgenden Punkten in Bezug auf J2EE/J2SE eingeschränkt:

1. Es sind weniger Klassenbibliotheken vorhanden. Das führt zu einem geringeren Funktionsumfang.
2. Die Virtuelle Maschine ist weniger leistungsfähig als die der J2EE/J2SE. Dies ist beispielsweise zurückzuführen auf das Fehlen eines Just-in-time-Compilers.
3. Es wird nicht notwendigerweise der komplette Sprachumfang von Java unterstützt.

Ein weiter Aspekt sind andere Philosophien und Schnittstellen, bspw. bei der Netzwerkprogrammierung oder der Gestaltung und Implementierung grafischer Oberflächen.

Zusätzlich erschwert das Fehlen einer einheitlichen, durchgängigen Spezifikation und einer umfassenden Referenzimplementierung den Einsatz von J2ME. Es gibt diesbezüglich mehrere, teilweise überschneidende, teilweise ausschließende Spezifikationen, die alle unter dem Begriff J2ME zusammengefasst werden.

3.4.4.8 Technische Aspekte

Hier sollen technische Aspekte der J2ME erläutert werden, die in vorangegangenen Kapitel kurz erwähnt worden sind.

3.4.4.9 Kilobyte Virtual Machine

Die Kilobyte Virtual Machine (KVM) wird für resourcenlimitierte Endgeräte verwendet. Ziel des Einsatzes der KVM ist es, essenzielle Eigenschaften von Java zu unterstützen und dabei mit einer Speicherkapazität von 160 bis 512 kByte auszukommen.

Als Basis der KVM dient die JVM. Jedoch führt die Resourcenknappheit zu Einschränkungen der JVM:

- Fließkomma-Arithmetik wie *float* und *double* wird hier nicht unterstützt, da die nötige Hardware hierzu meist nicht vorhanden ist und eine Softwarerealisierung zu rechen- und speicherintensiv ist.
- Das Java Native Interface (JNI) ist nicht vorhanden, da es zum einen zu umfangreich wäre und zum anderen das Sicherheitskonzept der J2ME unterwandern würde.
- Die KVM besitzt einen festen Classloader, der nicht ersetzt werden kann.
- Reflections sind nicht vorhanden.
- Nebenläufige Threads werden zwar unterstützt, jedoch keine Threadgruppen und Daemonthreads.
- Die GarbageCollection ist nur eingeschränkt vorhanden. Es gibt keine Finalisation.
- WeakReferences stehen nicht zur Verfügung
- Es gibt zwar den Mechanismus der Fehlerbehandlung, aber bei der Unterscheidung zwischen geprüften und ungeprüften Exceptions wird die Unterkategorie java.lang.Error der ungeprüften Ausnahmen nicht unterstützt (Schmatz, 2004).

Diese Limitierungen führen auch dazu, dass die Umsetzung auf PDAs nicht empfohlen werden kann, denn die Einschränkungen sind zu groß, zumindest in Hinblick auf Java. Ob eine andere Programmiersprache in diesem Fall besser geeignet ist, sei dahingestellt.

3.5 Quantität und Qualität

3.5.1 Verwaltung

Bei der Umsetzung der Analyse und des Entwurfs wurde darauf geachtet, die gewonnenen Erkenntnisse so weit wie möglich umzusetzen. Es sind jedoch auf Grund des Umfangs und des zentralen Gegenstands dieser Arbeit keine Untersuchungen gemacht worden, wie viele Personen mittels der dezentralen Clusteranalyse stabil verwaltet werden können. In den Versuchen habe ich mich lediglich auf maximal sieben Agenten beschränkt. Auch wurden keine Untersuchungen über das Laufzeitverhalten angestellt.

Bei einer möglichen Realisierung sollten diese beiden Punkten auf jeden Fall überprüft werden.

3.5.2 Evaluation

Mit dem nun erlangten Wissen und mit Blick auf das Beispielszenario aus Kapitel 1.4 wurde die Multiagentensystemgestützte Clusteranalyse prototypisch realisiert. Wir nehmen die Rolle eines uns aus Kapitel 2.6 wohl bekannten Gastes ein: Lilly.

3.5.2.1 Profil anlegen

Lilly hat nach dem Einchecken ein mobiles Endgerät bekommen und kann mittels des Personal Information Managements (PIM) die Profildaten verwalten. Das PIM ist durch einen SwingClient realisiert worden. Wenn Lilly ihr Profil dem System bekannt machen möchte, muss sie es anlegen. Dazu ruft sie den entsprechenden Eintrag „Anlegen“ im Untermenu „Profil“ auf (siehe Abbildung 3.26).

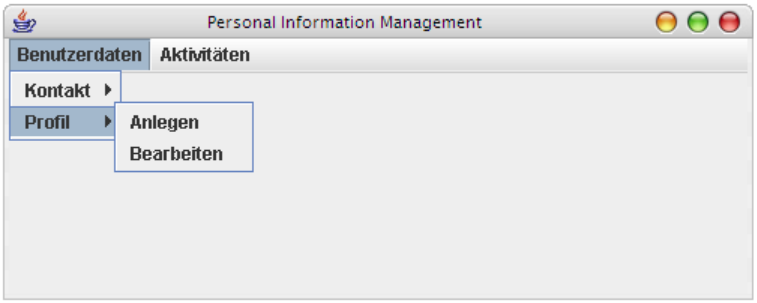

Abbildung 3.26: Profil anlegen

Es öffnet sich ein neues Fenster, in dem Lilly nun die Werte in die Profilvorlage eintragen kann. Hierzu muss sie allerdings sich selbstkritisch einschätzen (siehe Abbidung 3.27).

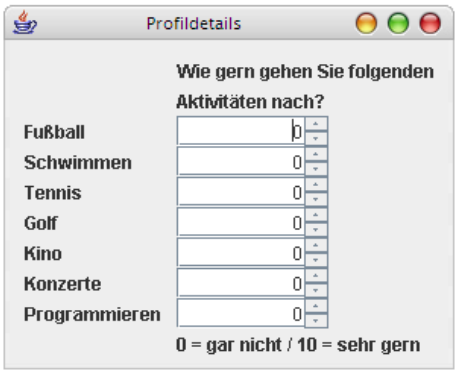

Abbildung 3.27: Ein neues Profil gemäß einer Vorlage

3.5.2.2 Profil bearbeiten

Nach entsprechender Eingabe kann das Profil jederzeit über den Eintrag „Bearbeiten“ im Untermenu „Profil“ des Personal Information Managements eingesehen und bearbeitet werden (siehe Abbildung 3.28).

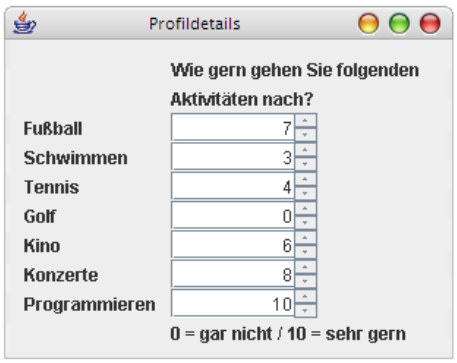

Abbildung 3.28: Das Profil bearbeiten bzw. einsehen

Neben der Profilverwaltung besteht auch die Möglichkeit der Verwaltung persönlicher Daten. Da diese jedoch bis auf den Namen des Benutzers keine Daten enthalten, soll diese Möglichkeit nicht näher betrachtet werden.

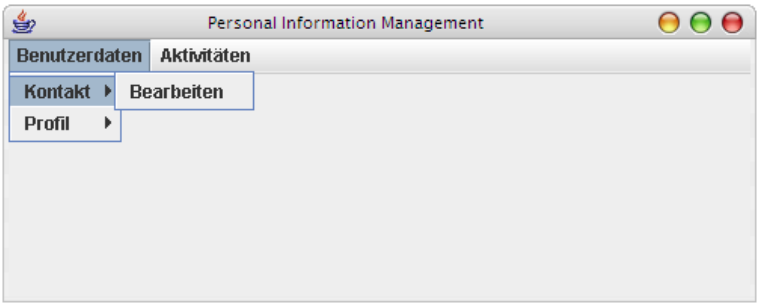

Abbildung 3.29: Kontaktdatenverwaltung

3.5.2.3 Einladung annehmen bzw. ablehnen

Man nehmen einmal an, der Ferienclub versende nun an alle Gäste eine Einladung zum Essen. Bei jedem mobilen Endgerät eines Gastes erscheint nun ein entsprechender Hinweis. Bei Lilly (und bei jedem anderen auch) sieht dieser Hinweis aus wie in Abbildung 3.30.

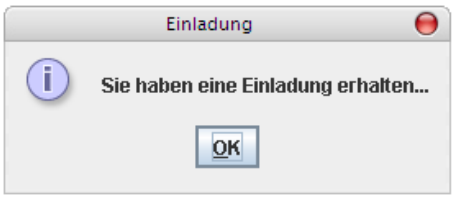

Abbildung 3.30: Hinweis auf erhaltene Einladung

Das PIM besitzt ein Menu „Aktivitäten“. Über den Eintrag „Einladungen“ kommt man zu einer Übersicht aktueller vorhandener, nicht angenommener Einladungen. Diese können hier eingesehen werden. Lilly hat hier die Möglichkeit, die Einladung anzunehmen oder abzulehnen, wie aus Abbildung 3.31 klar ersichtlich ist.

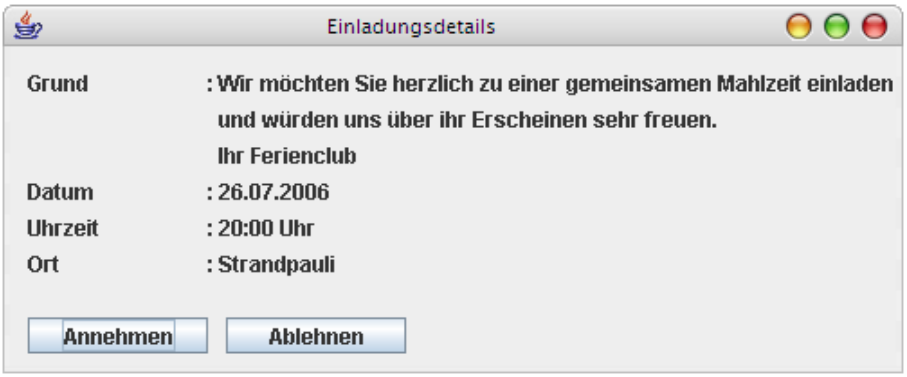

Abbildung 3.31: Die Einladung im Detail

3.5.2.4 Reservierung einsehen

Nach Annahme einer Einladung wird diese zu einer Reservierung. Auch dafür gibt es im Menu „Aktivitäten“ einen entsprechenden Eintrag. Über diesen kommt man zu einer Übersicht aktueller Reservierung, welche nun eingesehen werden können (siehe Abbildung 3.32).

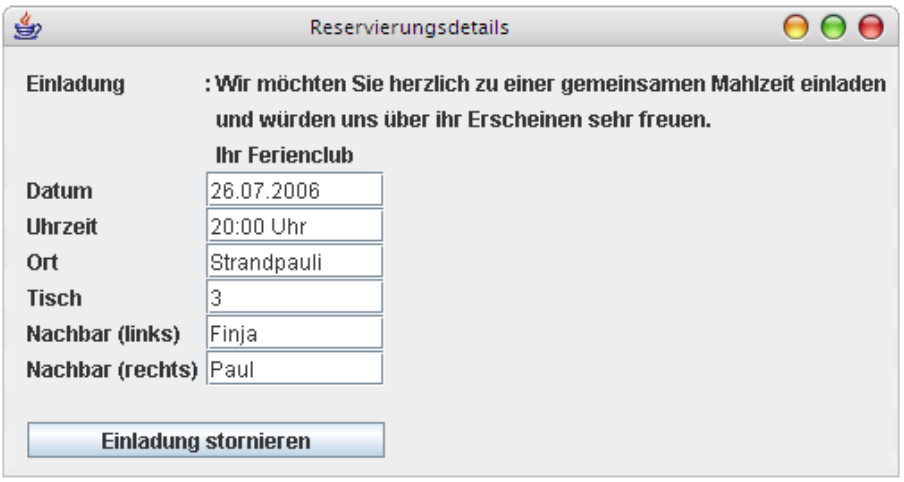

Abbildung 3.32: Die Reservierung im Detail

In dieser Detailansicht hat Lilly die Möglichkeit, eine Reservierung zu stornieren. In diesem Fall würde die Reservierung dann wieder zu einer nicht angenommenen Einladung und in entsprechender Übersicht vorhanden. Jedoch befinden sich in dieser Ansicht auch die Ergebnisse der Clusteranalyse. Man erkennt, zu welchem Cluster Lilly gehört und wer die jeweiligen Tischnachbarn sind.

3.6 Fazit

Zu Beginn hat man gesehen, worin der Unterschied der zentralen und der dezentralen Clusteranalyse liegt. Anschließend wurde ein Verfahren vorgestellt, das die Überführung den zentralen Fuzzy-C-Means-Algorithmus in eine dezentrale Variante desselbigen darstellt.

Doch dadurch, dass man nun in der Lage war, Gruppen zu bilden, entstanden neue Probleme hinsichtlich der räumlichen Positionierung der Individuen an einem Tisch und der Größe einer Gruppe. Die räumliche Positionierung wurde bei der Realisierung jedoch nicht berücksichtigt, da diese Diskussion eher zweitrangig ist. Elementar hingegen ist die Problematik der Gruppengröße, da diese direkt im Verfahren der Clusterbildung eine Rolle spielt. Alle Gedanken diesbezüglich wurde aufgegriffen und realisiert.

Anschließend ging es dann in die Tat über. Man hat gesehen, welche Entwurfsmuster bei der Realisierung eine Rolle spielen. Diese wurden im Prototypen auch vollständig umgesetzt. Ebenso wurde die modifizierte Agentenplattform komplett im Prototypen realisiert.

Dann wurde es ganz konkret mit in Form der verwendeten Hochsprache bei der Realisierung. Dabei wurde auch deutlich, woran eine sinnvolle Portierung des Systems auf einen

PDA letztendlich scheiterte. So befinden wir uns jetzt in der Position, diese Arbeit und (Butrynowski, 2005) zu resümieren, ein endgültiges Fazit zu ziehen und einen Ausblick auf die Weiterentwicklung des Prototypen zu geben.

4 Abschließende Bemerkungen

Am Ende dieser Arbeit soll der Inhalt noch einmal in Form einer Zusammenfassung und eines anschließenden Fazits rekapituliert werden. Anschließend soll ein Ausblick auf die Möglichkeiten des Einsatzes und der Erweiterung der Profilverarbeitung - speziell auf die dezentrale - gegeben werden.

4.1 Zusammenfassung

Ziel dieser Arbeit war die Entwicklung eines Verfahrens zur dezentralen Clusteranalyse auf Basis von Multiagentensystemen. Diese Clusteranalyse sollte am Beispiel eines gemeinsamen Abendessens einer größeren Gruppe von Personen durchgeführt werden. Dabei sollte die große Personengruppe in Untergruppen aufgeteilt werden, wobei sich die in einer Gruppe befindlichen Personen möglichst gut verstehen sollten. Dieses Ziel wurde erreicht. Die Bedingungen, unter denen das Ziel erreicht wurde, werden später erläutert.

Zu Beginn wurde eine allgemein bekannte Situation in einem Unternehmen dargestellt und implizite Probleme grob erwähnt. Anschließend erfolgte die Legitimation der Benutzung mobiler Endgeräte. Dabei wurde zunächst die Verbreitung moderner Informations- und Kommunikationstechnologie beleuchtet und eine Tendenz der Entwicklung dieser Verbreitung aufgezeigt. Anschließend kamen kritische Stimmen zu Wort. Ängste und Befürchtungen bezüglich des Einflusses der IKT auf die Gesellschaft und das Individuum wurden deutlich in Form der Vernetzungskritik. Jedoch stehen den Vernetzungskritikern die Aktivisten im Kontext der Sozialen Software gegenüber, die das genaue Gegenteil der Kritiker propagieren.

Danach wurde das in der Einleitung eher grob erwähnte Problem aufgeteilt in die dezentrale Natur, die Form der Unterstützung des menschlichen Individuums und die Clusteranalyse. Dadurch ergab sich ein Ziel, welches mit Beendigung dieser Arbeit erreicht worden ist: das Schaffen einer sozialen, verteilten Applikation zur Bildung von Personengruppen. Konkretisiert wurden Problem und Ziel an Hand eines Beispielszenarios, dass sich im Kontext eines gemeinschaftlichen Essen mehrerer Personen bewegte.

Die Analyse der Probleme brachte jedoch zu Tage, dass man sich mit derartigen Problemen in diversen Kontexten bewegt. Zwei davon wurden kurz erläutert: das ubiquitäre Computing und die Ambient Intelligence. Eine anschließende Anforderungsanalyse hinsichtlich der Problemstellung und der Zielsetzung brachte mehrere durch sie implizierte Vorstellungen zu Tage. Diese Vorstellungen gliederten sich in drei Bereiche.

Die allgemeinen Anforderungen aus der Informatik ließen sich durch Modularität, Wiederverwendbarkeit, Wartbarkeit, Erweiterbarkeit und Portabilität im Bereich der qualitativen Implementationsbewertung finden. Diese Ziele wurden in der Realisierung aufgegriffen und weitestgehend umgesetzt. Der Umfang der datenschutzrechtlichen Anforderungen hätte den Umfang dieser Arbeit gesprengt, darum wurden diese nur kurz erwähnt. Doch die Erwähnung sollte zeigen, dass man sich bei der prototypischen Entwicklung der Problematik zwar bewusst war, diese jedoch bewusst ausgeklammert hat. Letztendlich wurden Rollen und Anwendungsfälle aus dem Beispielszenario extrahiert und erläutert.

Im Kapitel 2.6 wurde es dann elementar. Es wurden Profile im Allgemeinen erklärt. Und da es ein Ziel war, in sich harmonische Gruppen von Personen zusammenzustellen, musste erst einmal geklärt werden, was harmonisch in diesem Sinne überhaupt heißt. Dazu wurden zwei konträre Sichtweisen vorgestellt und die in dieser Arbeit gültige belegt. Hier gilt: „Gleich und gleich gesellt sich gern“.

Nach der allgemeinen Einführung in Profile sind diese in Bezug auf deren Verwendung in dieser Arbeit konkretisiert worden. Zur Darstellung und Verarbeitung wurden die Profile in den Bereich der Mathematik überführt. Anschließend wurde gezeigt, wie Ähnlichkeit mathematisch berechnet werden kann. Auch wurde eine Form der Manipulation von Ähnlichkeit vorgestellt, die jedoch nicht realisiert wurde.

Für die impliziten graphentheoretischen Probleme hinsichtlich vollständiger Graphen gibt es mehrere Lösungsansätze. In dieser Arbeit wurde ein Ansatz mit Neuronalen Netzen und ein Ansatz aus der Clusteranalyse vorgestellt. Da die Realisierung der Clusteranalyse Gegenstand dieser Arbeit ist, wurde dieser Teil eingehend untersucht. Eine kritische Betrachtung der mathematischen Umsetzung von Profilen ergab, dass der Einsatz von Fuzzy-Logik an dieser Stelle angebracht ist. Untermauert wird dies durch die Tatsache, dass die Profile dadurch entstehen, dass sich Personen selbst einschätzen müssen. Somit ist das zu Grunde liegende Wissen extrem vage, was für den Einsatz der Fuzzy-Logik spricht. Da es in diesem Bereich ebenfalls eine Form der Clusteranalyse gibt, wurde diese aufgegriffen.

Die Fuzzy-Clusteranalyse und die Fuzzy-Logik zur Überprüfung der Ähnlichkeit von Profilen wurden in einen dezentralen Algorithmus überführt und umgesetzt. Dabei wurden deutlich Unterschiede und Gemeinsamkeiten von zentraler und dezentraler Clusteranalyse hervorgehoben. Bei der Umsetzung traten Probleme hinsichtlich der Anordnung von Personen an einem Tisch und der Größe von Gruppen zu Tage. Wie nahe Personen zusammen sitzen

dürfen und wie groß Gruppen im Minimum beziehungsweise im Maxiumum sein dürfen, waren Fragen, denen ich mich hier angenommen habe. Die Lösung wurde an entsprechender Stelle vorgestellt.

Diese Fragestellungen beinhalten allerdings auch das Vorhandensein verwertbarer Informationen. Da es hier um Profile geht, wurde kurz erklärt, welche Möglichkeiten es gibt, die Profile mit Informationen zu füllen.

Die verteilte Natur der Problemstellung setzte eine kurze Einführung in die Verteilten Systeme voraus, damit sich anschließend der verteilten Künstlichen Intelligenz in Form von kooperierenden Softwareagenten - auch Multiagentensysteme genannt - gewidmet werden konnte. Dabei ging es von einzelnen Softwareagenten über kooperierende Agenten bis hin zur Schwarmintelligenz. Die Umwelt von Softwareagenten und die Möglichkeiten der Wissensdarstellung und Kommunikation wurden ebenso wie ihre prinzipielle Arbeitsweise vorgestellt. Anschließend gab es einen kleinen Exkurs in das Foschungsprojekt Sozionik, welches sich der Erforschung menschlicher Verhaltensweisen durch Multiagentensysteme widmet und dabei auf bidirektionale Verbesserung baut: einerseits bezüglich der Verbesserung der Implementation von Agenten, andererseits in Bezug auf humanoide Sozialstrukturen.

Bei dem Design wurden dann nach einer Analyse des generellen Ablaufs hinsichtlich der Teilnahmebestätigung zum gemeinschaftlichen Essen die zu Grunde liegende Architektur vorgestellt. Bei dem Entwurf stieß man auf bekannte Probleme und entsprechende Lösungen wurden in Form von Entwurfsmustern wie dem Model-View-Controller-Muster, dem Beobachter-Muster oder dem PlugIn-Muster vorgestellt. Ebenfalls wurde eine bekannte Agentenplattform dieser Arbeit angepasst und implementiert. Die verwendete Programmiersprache Java und die Probleme hinsichtlich der Portierung auf mobile Engeräte im Kontext der Java 2 Mirco Edition wurden behandelt. Durch diese Probleme war eine sinnvolle Portierung auf einen PDA leider nicht möglich.

In Anbetracht der gesetzten Ziele kann ich an dieser Stelle sagen, dass es gelungen ist, eine verteilte Anwendung zu realisieren, die den Ansprüchen einer sozialen Software genügt und eine Clusteranalyse entsprechend der Kriterien ordnungsgemäß durchführt. Da die Initialisierung jedoch über einen Server läuft, ist eine 100%ige Dezentralisierung leider nur zu 95% relisiert worden. Eine andere Möglichkeit gab es unter Berücksichtigung der verwendeten Verfahren jedoch nicht.

4.2 Fazit

Da (Butrynowski, 2005) in seinem Ausblick konkrete Vorstellungen gehabt hat, die in dieser Arbeit diskutiert wurden, sollen diese hier kurz aufgegriffen werden und mit den Ergebnissen dieser Arbeit gegenübergestellt werden.

Dezentral vs. zentral

Im Hinblick auf die dezentrale Clusteranalyse kann gesagt werden, dass der Einsatz von Agenten in diesem Kontext durchaus eine nennenswerte Alternative zur zentralen Clusteranalyse darstellt.

Die implizite Zielsetzung der Transparenz des Computers in der indirekten Mensch-zu-Mensch-Kommunikation konnte jedoch nicht in dem erhofften Umfeld realisiert werden. Ausgehend von dem Einsatz auf Desktops und der sukzessiven Dekrementierung der Größe der Rechnerarchitektur, haben Tests in folgenden Umgebungen stattgefunden:

- Desktop-Computer
- Laptop
- PDA

Der Einsatz der Applikation auf dem PDA implizierte einen Wechsel der Virtuellen Maschine bzgl. JAVA. Leider ist die Laufzeitumgebung für mobile Endgeräte zu stark beschränkt, als dass man einen sinnvollen Einsatz in dem gegebenen Kontext zustimmen könnte. Größere PDAs - zum Beispiel Sharp Zaurus CLxxx - oder auch Subnotebooks verfügen jedoch über eine geeignete technische Infrastruktur zur Realisierung der Anwendung. Der Trend der Entwicklung mobiler Endgeräte in Bezug auf Leistungsfähigkeit zeigt jedoch, dass eine sinnvolle Umsetzung der Anwendung auf mobile Endgeräte in naher Zukunft durchaus realisierbar ist.

Die Frage nach der Leistungsfähigkeit kann sowohl für die zentrale als auch für die dezentrale Clusteranalyse gleich beantwortet werden. Da jedoch ein Teil der Daten zentral gehalten werden sollte, da die Clienten nicht unbedingt über große Speicherkapazität verfügen, ist eine Symbiose der beiden Varianten anzustreben. Welche Ausprägung dies annehmen kann, sollte separat untersucht werden.

Sicherheit

Im Kapitel 2.4 wurde bereits erwähnt, dass die Sicherheit in dieser Arbeit wenig Berücksichtigung fand. Jedoch ist im Vergleich zur zentralen Lösung mehr Sicherheit in Bezug auf die Profildaten gegeben. Diese werden nicht mehr zentral gehalten und können somit nur lokal eingesehen werden. So lange mobile Endgeräte durch ein Passwort geschützt sind, kann auch niemand die Daten des inzidenten Benutzers einsehen. Ist man jedoch am Endgerät eingelogged, so ist die Applikation nicht zusätzlich durch eine wie auch immer geartete Form der Authentifizierung geschützt. Aber es haben sich mittlerweile andere Arbeiten mit dem Thema Sicherheit in dem Umfeld des Ferienclubs beschäftigt, so dass sich hier sicher eine adäquate Lösung finden lässt (Lüpke, 2004; Elvers, 2004; Bartnik, 2005).

Man sollte an dieser Stelle zumindest über eine Form der Benutzer- und Rechteverwaltung nachdenken.

Persistenz

In der zentralen Lösung wurden die Daten nicht gespeichert. Die dezentrale Lösung „sichert“ die Daten in einer Datei, jedoch sollte man über den Einsatz mobiler Datenbanken nachdenken. Mobile Datenbanken wurden ebenfalls in einer Arbeit an der Hochschule für Angewandte Wissenschaften Hamburg behandelt. Diese Arbeit kann als Ansatz zur Erweiterung der Applikation mittels mobiler Datenbanken dienen.

4.3 Ausblicke

Die in dieser Arbeit vorgestellte (realisierte) Form der dezentralen Clusteranalyse kann als eigenständige Applikation umgesetzt werden oder als Basis für weitere Untersuchungen dienen. Ein zentraler Gegenstand meiner Untersuchungen ist die Erstellung und Verwaltung von Profilen gewesen. Je länger man sich mit dem Thema der Profilierung beschäftigt, desto mehr wird klar, dass es im Grunde zwei Formen der Profilierung gibt.

Statische Profilierung

Unter statischer Profilierung versteht man im Grunde die Art und Weise der Profilverwaltung, wie sie in dieser Arbeit angewendet wurde. Unter statischer Profilierung versteht eine Form expliziten Informationsgewinnung, da das vorhandene Profil nur durch den Benutzer initialisiert und aktualisiert werden kann.

Sowohl bei der Analyse als auch bei Design und Realisierung wurde statische Profilierung angewandt. Man muss sich nur einmal die Tatsache in Erinnerung rufen, dass in Kapitel 2.6 die Eingabe und Aktualisierung eines Profils durch den inzidenten Benutzers explizit gefordert wird.

Dynamische Profilierung

Der Nachteil der statischen Profilierung ist, dass eine Transparenz des Computers im Kontext der indirekten Mensch-zu-Mensch-Kommunikation nie erreicht werden kann, da der inzidente Benutzer immer gefordert ist. Aber auch im Kontext der Profilierung ist der Stellenwert der Mensch-zu-Maschine-Kommunikation zu groß. In diesem Kontext kann man eine Transparenz des Computers als Medium der Profilverwaltung jedoch zu einem sehr hohen Grad erreichen. Man kann sogar absolute Transparenz erreichen, wenn man sich an C.G.Jung und dem Begriff des Archeypen erinnert.

C.G.Jung und der Archetypus

Carl Gustav Jung war Schüler von Sigmund Freud und Gründer der analytischen Psychologie. Die analytische Psychologie wird auch als komplexe Psychologie bezeichnet und ist eine Weiterentwicklung der Psychoanalyse.

Ein Begriff, der häufig in der Lehre von Jung erwähnt wird, ist das „Kollektive Unbewusste". „[...]Dies besteht aus ererbten Grundlagen der Menschheitsgeschichte. Auf ihm beruhen alle entwicklungsgeschichtlich jüngeren Persönlichkeitsstrukturen, wie etwa dem Ich. Im kollektiven Unbewussten manifestieren sich Archetypen. Archetypen sind universell vorhandene Urbilder in der Seele aller Menschen, unabhängig von ihrer Geschichte und Kultur. Dazu zählen Gegenstände und Lebewesen aus der Umwelt wie etwa Bäume oder Bären. Die Existenz der Urbilder konnte Jung mit seinen Mitarbeitern nachweisen, indem er Menschen, die nachweislich noch nie einen Baum oder einen Bären gesehen hatten, und auch nicht mit Massenmedien in Berührung gekommen waren, ihre Träume malen ließ. Darauf malten sie deutlich Bäume und Bären. Jung recherchierte Jahrtausende altes Material aus vielen Kulturen und stellte in den Darstellungen fest, dass bestimmte Bilder, Motive und Symbole immer wieder auftauchten, unabhängig voneinander. Andere Urbilder treten dem Menschen in seinen Träumen vor Augen. Dazu gehören der „Schatten", „Anima" und „Animus", der oder die alte Weise, das Mandala, der Abstieg der Seele zum Wasser, der Abstieg ins Totenreich, das Numinose und andere.

Archetypen sind „Energiekomplexe", die besonders in Träumen, Neurosen und Wahnvorstellungen ihre Wirkmacht entfalten. Jung erklärt eine Psychose, die unter anderem dann entstehen kann, wenn eine Neurose nicht behandelt wird, als Überhandnehmen des Unbewussten, das sich des Bewusstseins bemächtigt, um dessen Einstellung zu korrigieren und das Individuum auf dem Weg zur Ganzwerdung zu befreien. Die nun „symbolisch" wirksamen Archetypen zielen darauf ab, die Gesamtpersönlichkeit wieder ins Lot zu bringen, indem sie urzeitliche, durch Numinosität sehr attraktive Zielbilder ins Bewusstsein aufsteigen lassen. Diese Bilder und die Beschäftigung der Seele mit ihnen haben die Aufgabe, der Persönlichkeit eine fundamentale Balance zurückzugeben, Sinn und Ordnung zu stiften. Sie manifestieren sich daher in symbolischen Bildern universeller Gültigkeit, die einen beträchtlichen Anteil am Leben eines jeden haben. Das Selbst ist das Zentrum der Persönlichkeit. In ihm werden alle gegenläufigen Teile der Persönlichkeit zusammengefasst und vereinigt. Es ist das Ziel des lebenslangen Individuationsprozesses, der im wesentlichen daraus besteht, möglichst große Teile des Unbewussten dem Bewusstsein einzugliedern. Die „Individuation" setzt immer neue und umfassendere Anpassungsleistungen der Persönlichkeit voraus und in Gang. Er findet auf der „Ich-Selbst-Achse" statt.[...]" (Wikipedia 2005).

Auch wenn dieser Ansatz sehr umstritten ist, kann er dennoch als Grundlage für weitere Forschungen dienen, denn wenn man nun jedes Profil mittels eines Archetypen initialisiert, entfällt die in Kapitel 2.6 geforderte Initialisierung durch den Benutzer.

Diese Form wird als dynamische Profilierung bezeichnet. Unter dynamischer Profilierung versteht man eine extreme Form der impliziten Informationsgewinnung. Der inzidente Benutzer eines Profils hat quasi überhaupt keine direkte Einflussmöglichkeit auf die Ausprägung der charakteristischen Attribute. Man kann sich dynamische Profilierung in mehreren Ausprägungen vorstellen. Als Basis sei wieder die Benutzung eines PDAs zur Verwaltung der Termine und zum Einkauf im Internet etc. vorausgesetzt. Die Profilierung greift hier an mehreren Stellen. Es könnte eine Analyse der Termine stattfinden, um festzustellen, wie Arbeits- und Freizeitgewohnheiten ausfallen. Wenn die Form der Aktivitäten ebenfalls mit aufgeführt werden, kann man diese auch genau analysieren. Es gäbe folgende Möglichkeiten:

Sparte	**Gegenstand der Analyse**
Sport	Welcher Sportkategorie wird nachgegangen? Selbstverteidigung, Mannschaftsport, Schießen, Schach? Diese können weiterhin auf ein mögliches wie auch immer geartetes Potenzial hin untersucht werden.
Arbeit	Welcher Art der Beschäftigung wird nachgegangen? Mögliche Position?
Kino	Welche Filme werden wo gesehen? Ausprägung?
Einkäufe	Was wird wo gekauft?
Mailverkehr	Wem wird geschrieben? Gegenstand des Mailverkehrs?
etc...	

Tabelle 4.1: Möglichkeiten der Terminanalyse

Man sieht also, dass die Granularität der Analyse beliebig fein sein kann, je nachdem, wonach man „sucht“.

Anwendung würde diese Form in diversen Kontexten finden, sei es bei kriminalistischer Arbeit im Sinne eines Profilers. Man könnte diese Form der Profilierung auch bei der Terrorismusbekämpfung einsetzen.

Allgemein ist jedoch bei der dynamischen Profilierung größte Vorsicht geboten. Das Profil beziehungsweise dessen Ausprägungen werden immer nur durch logische oder vage Schlüsse modifiziert. Die Regeln der Modifizierung erfolgen jedoch individuell je nach Ansicht der Entwickler.

So könnte man ein Beispiel im Kontext der Terrorismusbekämpfung betrachten. Nehmen wir eine Person, die sich ab und zu mal in einer Moschee blicken lässt, regelmäßig zum Aikido geht und gern Actionfilme ansieht. Man hat es quasi mit einem Individuum zu tun, deren religiöse Weltanschauung vom Islam geprägt ist und deren Kenntnis der Kampfkunst und der Vorliebe für Actionfilme eine klaren Hang zur Gewalt offenbart. So könnte also mittels der dynamischen Profilierung diese Person als vermeintlich gefährliches Subjekt klassiiziert werden. Eine Einreise in die USA wäre nicht mehr möglich.

Man sollte an dieser Stelle sagen, dass beiden Formen in ihrer reinen Anwendung ungeeignet sind. Vielmehr sollte man versuchen, eine hybride Form der Profilierung zu entwickeln, die implizit erfolgt und direkten Einfluss ermöglicht.

Doch soll der Blick nicht nur in Richtung Zukunft und eine mögliche Weiterentwicklung gerichtet sein. Die in dieser Arbeit entwickelte Applikation hat vielfältige Einsatzmöglichkeiten. Neben den in Kapitel 1 erwähnten Möglichkeiten in Personalvermittlung und Teamkonfiguration ist auch ein Einsatz im Kontext von *GroupWare* möglich.

Doch auch größere Firmen machen sich Gedanken über den Einsatz von ubiquitärem bzw. Ambient Computing. In Bezug auf das ubiquitäre Computing bzw. die Ambient Intelligence geht die Version von Hewlett-Packard sogar noch einen Schritt weiter (siehe auch das *Cooltown*-Projekt von HP).

Literaturverzeichnis

[A.K.Caglayan und C.G.Harrison 1998] A.K.CAGLAYAN ; C.G.HARRISON: *Intelligente Software-Agenten*. Hanser, 1998. – ISBN 3-446-19269-7

[Babic 2003] BABIC, Alexander: *Entwicklung einer profilverarbeitenden ubiquitären Anwendung*, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2003. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/diplom/babic.pdf

[Bacher 1996] BACHER, Johann: *Clusteranalyse*. Oldenbourg, 1996. – ISBN 3-486-23760-8

[Bartnik 2005] BARTNIK, Roman: *Sicheres WLan im Ferienclub*, Hochschule für Angewandte Wissenschaften Hamburg, Studienarbeit, 2005. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/studien/bartnik.pdf

[Bigus und Bigus 2001] BIGUS, Joseph ; BIGUS, Jennifer: *Intelligente Agenten mit Java programmieren*. Addison-Wesley, 2001. – ISBN 3-8273-1841-6

[Böhm 2002] BÖHM, Oliver: *Java Software Engineering unter Linux*. SuSE Press, 2002. – ISBN 3-935922-53-1

[Bresch 2004] BRESCH, Marco: *Erweiterung des Java Server Faces Frameworks f̈ur den Einsatz in mobilen Anwendungen*, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2004. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/diplom/bresch.pdf

[Brody 1994] BRODY, Jane E.: Die meisten Menschen suchen sich einen Partner, der ihnen ähnlich ist. In: *Welt am Sonntag* 51 (1994), Dezember, S. 32

[Butrynowski 2005] BUTRYNOWSKI, Christian: *Entwicklung eines Systems zur Clusteranalyse von Benutzerprofilen*, Hochschule für Angewandte Wissenschaften Hamburg, Studienarbeit, 2005. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/studien/butrynowski.pdf

[CP Exchange 2005] IDEALLIANCE (Hrsg.): *CPExchange Standards*. – URL http://www.cpexchange.org/standard/. – Global standards for privacy-enabled customer data exchange

[Dawson 2003] DAWSON, Christian W.: *Computerprojekte im Klartext*. Pearson Studium, 2003. – ISBN 3-8273-7067-1

[DeMarco und Lister 1999] DEMARCO, Tom ; LISTER, Tomothy: *Wien wartet auf Dich - Der Faktor Mensch im DV-Management*. Carl Hanser Verlag München Wien, 1999. – ISBN 3-446-21277-9

[Döring 1996a] DÖRING, Nicola: Führen Computernetze in die Vereinsamung? In: *Zeitschrift für angewandte Sozialpsychologie* 27 (1996), September, Nr. 3

[Döring 1996b] DÖRING, Nicola: Soziale Netze + Computernetze = glückliche Verbindungen? In: *Forum Medienethik* (1996), September, Nr. 2, S. 45–51

[Drosdowski u. a. 1997] DROSDOWSKI, Prof. Dr. Dr. h.c. Günther (Hrsg.) ; SCHOLZE-STUBENRECHT, Dr. W. (Hrsg.) ; WERMKE, Dr. M. (Hrsg.): *Das Fremdörterbuch*. Duden, 1997. – ISBN 3-411-04056-4

[Eilebrecht und Starke 2004] EILEBRECHT, Karl ; STARKE, Gernot: *Patterns kompakt*. Spektrum Akademischer Verlag, 2004. – ISBN 3-8274-1443-1

[Elvers 2004] ELVERS, Sven: *Eine effiziente Sicherheitsarchitektur für verteilte Systeme*, Hochschule für Angewandte Wissenschaften Hamburg, Bachelorarbeit, 2004. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/bachelor/elvers.pdf

[Ferber 2001] FERBER, Jacques: *Multiagentensysteme*. Addison-Wesley, 2001. – ISBN 3-8273-1679-0

[Frank 2002] FRANK, Hubert: *Fuzzy Methoden in der Wirtschaftsmathematik*. Vieweg Verlag, 2002. – ISBN 3-528-03195-6

[Friedrich 1997] FRIEDRICH, Alfred: *Logik und Fuzzy-Logik*. expert Verlag, 1997. – ISBN 3-8169-1405-5

[Gamma u. a. 1996] GAMMA, Erich ; HELM, Richard ; JOHNSON, Ralph ; VLISSIDES, John: *Entwurfsmuster*. Addison-Wesley, 1996. – ISBN 3-89319-950-0

[Goossens u. a. 2000] GOOSSENS, Michel ; MITTELBACH, Frank ; SAMARIN, Alexander: *Der LaTeX -Begleiter*. Addison-Wesley, 2000. – ISBN 3-8273-1689-8

[Görz u. a. 2003] GÖRZ, G. ; ROLLINGER, C.-R. ; (HRSG.), J. S.: *Handbuch der Künstlichen Intelligenz*. Oldernbourg, 2003. – ISBN 3-486-27212-8

[Günther 2002] GÜNTHER, Karsten: *LaTeX GE-PACKT*. mitp, 2002. – ISBN 3-8266-0785-6

[Heinsohn und Socher-Ambrosius 1999] HEINSOHN, Jochen ; SOCHER-AMBROSIUS, Rolf: *Wissensverarbeitung*. Spektrum Akademischer Verlag, 1999. – ISBN 3-8274-0308-1

[Hewlett-Packard 2004] HEWLETT-PACKARD (Hrsg.): *Making Cooltown real*. – URL http://www.cooltown.com/cooltown/

[Höppner u. a. 1997] HÖPPNER, Frank ; KLAWONN, Frank ; KRUSE, Rudolf: *Fuzzy-Clusteranalyse*. Vieweg Verlag, 1997. – ISBN 3-528-05543-X

[Hörauf 2001] AKADEMIE.DE (Hrsg.): *Hewlett-Packard Cooltown Project*. 2001. – URL http://www.informatik.uni-mannheim.de/pi4/lectures/ws0102/UbiquitousComputing/cooltown_arbeit.pdf. – Universität Karlsruhe - Institut für Telematik, Seminararbeit zum Seminar Ubiquitäre Systeme

[Hunt und Thomas 2003] HUNT, Andrew ; THOMAS, David: *Der Pragmatische Programmierer*. Hanser, 2003. – ISBN 3-446-22309-6

[Jaanineh und Maijohann 1996] JAANINEH, Georg ; MAIJOHANN, Markus: *Fuzzy-Logik und Fuzzy-Control*. Vogel Verlag, 1996. – ISBN 3-8023-1535-9

[Kennedy u. a. 2001] KENNEDY, James ; EBERHARDT, Russell C. ; SHI, Yuhui: *Swarm Intelligence*. Morgan Kaufmann, 2001. – ISBN 1-55860-595-9

[Kohm und Morawksi 2003] KOHM, Markus ; MORAWKSI, Jens-Uwe: *KOMA-Script*. dante, 2003. – ISBN 3-936427-45-3

[Kollakowski 2004] KOLLAKOWSKI, Malte: Mobile Aktivitäten. In: *Der Entwickler* (2004), April, Nr. 4.04, S. 15–20

[Kruse 2000] KRUSE, Otto: *Keine Angst vor dem leeren Blatt*. campus concret, 2000. – ISBN 3-593-36659-2

[Kühnel 2001] KÜHNEL, Ralf: *Agentenbasierte Softwareentwicklung*. Addison-Wesley, 2001. – ISBN 3-8273-1739-8

[Kurose und Ross 2002] KUROSE, James F. ; ROSS, Keith W.: *Computernetze*. Pearson Studium, 2002. – ISBN 3-8273-7017-5

[Lamport 1995] LAMPORT, Leslie: *Das LaTeX Handbuch*. Addison-Wesley, 1995. – ISBN 3-89319-826-1

[Langham 2005] LANGHAM, Matthew: Connecting People. In: *SERVER Magazin* 1 (2005), S. 49–52

[von Lüde u. a. 2004] LÜDE, Rolf von ; MOLDT, Daniel ; VALK, Rüdiger: *Sozionik - Modellierung soziologischer Theorie*. LIT, 2004. – ISBN 3-8258-5980-0

[Luger 2001] LUGER, George F.: *Künstliche Intelligenz*. Pearson Studium, 2001. – ISBN 3-8273-7002-7

[Lüpke 2004] LÜPKE, André: *Entwurf einer Sicherheitsarchitektur für den Einsatz mobiler Endgeräte*, Hochschule für Angewandte Wissenschaften Hamburg, Diplomarbeit, 2004. – URL http://users.informatik.haw-hamburg.de/~ubicomp/arbeiten/diplom/luepke.pdf

[Maas und Klauck 1999] MAAS, Christoph ; KLAUCK, Christoph: *Graphentheorie und Operations Research für Studierende der Informatik*. Wißner, 1999. – ISBN 3-89639-178-X

[Mattern 2005] MATTERN, Friedemann (Hrsg.): *Pervasive Computing - Ubiquitous Computing*. – URL http://www.inf.ethz.ch/vs/publ/papers/UbiPvCSchlagwort.pdf. – Veröffentlichung an der Eidgenössischen Technischen Hochschule Zürich

[Möller 2005] MÖLLER, Erik: *Die heimliche Medienrevolution - Wie Weblogs, Wikis und freie Software die Welt verändern*. Heise, 2005. – ISBN 3-936931-16-X

[netlexikon 2005] AKADEMIE.DE (Hrsg.): *netlexikon*. – URL http://www.lexikon-definition.de/

[Nilsson 1998] NILSSON, Nils J.: *Artificial Intelligence: A New Synthesis*. Morgan Kaufmann, 1998. – ISBN 1-55860-535-5

[P3P 2005] W3C (Hrsg.): *Platform For Privacy Preferences (P3P) Project*. – URL http://www.w3.org/P3P/

[PM 2004] Gleich und gleich gesellt sich gern. Stimmt das? In: *PM Fragen + Antworten* 8/04 (2004), August

[Poenicke 1988] POENICKE, Klaus: *Wie verfaßt man wissenschaftliche Arbeiten?* Duden, 1988. – ISBN 3-411-02751-7

[Pötzsch u. a. 2003] PÖTZSCH, Olga ; KORTH, Birgit ; SCHNORR-BÄCKER., Dr. S.: *Informationstechnologie in Haushalten - Ergebnisse einer Pilotstudie für das Jahr 2002*. Statistisches Bundesamt - Pressestelle, Wiesbaden, 2003. – URL http://www.destatis.de/informationsgesellschaft/d_home.htm

[Rötzer 1999] RÖTZER, Florian (Hrsg.): *Ein neuer Standard soll Kundendaten zusammenführen*. Nov. 1999. – URL http://www.heise.de/newsticker/meldung/6970

[Scheppach 1988] SCHEPPACH, Joseph: Sprichwörter: Viele haben einen wahren Kern. In: *PM* 9/88 (1988), September

[Schirru 2004] SCHIRRU, Rafael (Hrsg.): *Trust, Reputation, Privacy*. Jul. 2004. – URL http://wwwdvs.informatik.uni-kl.de/courses/seminar/SS2004/rschirrub.pdf. – Universität Karlsruhe - Lehrgebiet Datenverwaltungssysteme, Seminararbeit zum Seminar Grundlagen webbasierter Informationssysteme

[Schmatz 2004] SCHMATZ, Klaus-Dieter: *Java 2 Micro Edition - Entwicklung mobiler Anwendungen mit CLDC und MIDP*. dpunkt.verlag, 2004. – ISBN 3-89864-271-2

[Sixtus 2005] SIXTUS, Mario: Das Web sind wir. In: *Technology Review* (2005), Juli, Nr. 7, S. 44–52

[Streitz und Nixon 2005] STREITZ, Norbert ; NIXON, Paddy: The Disappearing Computer. In: *Communications of the ACM* 48 (2005), März, Nr. 3, S. 33–35

[Vogt 2001] VOGT, Carsten: *Betriebssysteme*. Spektrum Akademischer Verlag, 2001. – ISBN 3-8274-1117-3

[Weiß und Jakob 2005] WEISS, Gerhard ; JAKOB, Ralf: *Agentenorientierte Softwareentwicklung*. Springer, 2005. – ISBN 3-540-00062-3

[Wikipedia 2005] : *Wikipedia - Die freie Enzyklopädie*. – URL http://de.wikipedia.org/wiki/

[Wise 2004] WISE, Edwin: *Hands-on AI with JAVA*. McGraw Hill, 2004. – ISBN 0-07-142496-2

[Wooldridge 2002] WOOLDRIDGE, Michael: *MultiAgent Systems*. Wiley, 2002. – ISBN 0-471-49691-X

A Anhang

A.1 Versuche und Ergebnisse

Bei der Fertigstellung von Kapitel 3 gab es einige Experimente. Screenshots davon hat man bereits in Kapitel 3.5 gesehen. Diesen Experimenten lag das Szenario aus Kapitel 3.5.2 zu Grunde. Dieses Szenario wurde mit den Profilen aus Kapitel 3.2 experimentell umgesetzt. Dort waren die folgenden Personen zu einem gemeinsamen Essen eingeladen worden.

Person	**Clusterzugehörigkeit**	
	C_1	C_2
Finja	0,7	0,6
Lilly	0,7	0,7
Mia	0,6	0,1
Paul	0,6	0,1
Tom	0,3	0,8
Louis	0,2	0,9

Tabelle A.1: Clusterzugehörigkeiten

Jede dieser Personen hat die Einladung angenommen. Bei der Clusterbildung wurden mehrere Versuche unter jeweils unterschiedlichen Bedingungen gemacht. Nachfolgend werden die Versuche und ihre Ergebnisse kurz vorgestellt.

A.1.1 Versuch 1: Clustergröße ohne obere Grenze

Dies war der Initialversuch. Es wurde eine untere Clustergröße von zwei Personen festgelegt. Es gab keine obere Grenze. Der letzte Zustand, in dem sich das System befand, enthielt folgende Cluster:

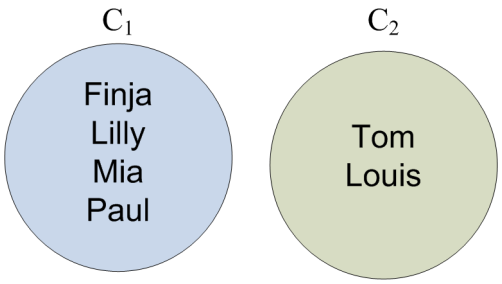

Abbildung A.1: Clusterbildung des 1.Versuchs

A.1.2 Versuch 2: Erhöhung der unteren Grenze

Im nächsten Versuch wurde die Mindestanzahl an Personen, die sich in einem Cluster befinden müssen, auf drei erhöht. Das führte zu dem Ergebnis, dass Lilly, die offensichtlich zu beiden Clustern den gleichen Zugehörigkeitsgrad besitzt, aus dem Cluster C_1 in den Cluster C_2 gewechselt ist.

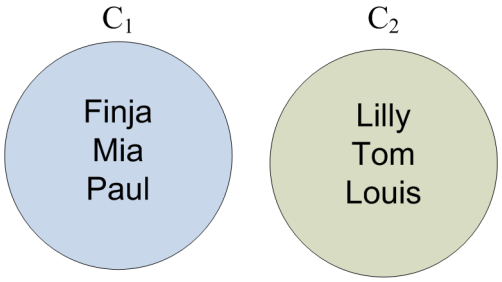

Abbildung A.2: Clusterbildung des 2.Versuchs

A.1.3 Versuch 3: Einführung einer oberen Grenze

Der letzte Versuch beinhaltete die Einführung einer oberen Grenze von vier Personen. Die untere Grenze wurde dabei wieder auf zwei Personen herabgesetzt. Damit dieser Versuch vollzogen werden konnte, kam ein neuer Gast in den Ferienclub: Duke. Es waren demnach die folgenden Gäste in den Club involviert:

Person	**Clusterzugehörigkeit**	
	C_1	C_2
Finja	0,7	0,6
Lilly	0,7	0,7
Mia	0,6	0,1
Paul	0,6	0,1
Tom	0,3	0,8
Louis	0,2	0,9
Duke	0,9	0,1

Tabelle A.2: Clusterzugehörigkeiten incl. des neues Gastes

Die Grundkonfiguration dieses Versuch stimmt also mit der des ersten Versuchs überein. Der erste Versuch würde jedoch folgendes Ergebnis liefern:

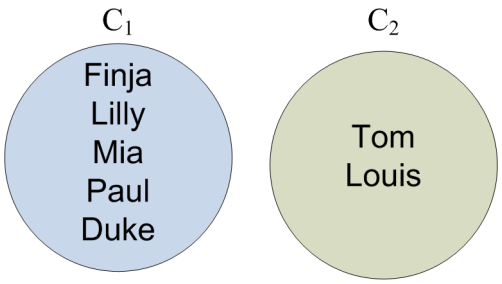

Abbildung A.3: Clusterbildung des 3.Versuchs ohne obere Grenze

Die Einführung einer oberen Grenze hatte jedoch zur Folge, dass sich der Cluster C_1 geteilt hat. Damit ergaben sich folgende Cluster:

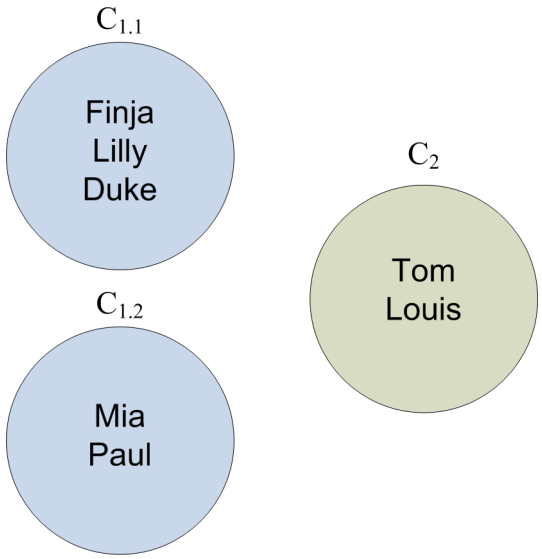

Abbildung A.4: Clusterbildung des 3.Versuchs mit oberer Grenze

Glossar

Agenten *sing. Agent*; Im Wirtschaftsauftrag ist die Bezeichnung Agent in Deutschland veraltet und durch den Handelsvertreter oder Makler ersetzt. Der Handelsvertreter ist selbständiger Kaufmann, der damit beauftragt ist, für einen anderen Unternehmer (Anbieter) Geschäfte zu vermitteln oder in dessen Namen abzuschließen (Wikipedia 2005).

Clusteranalyse Unter Clusteranalyse versteht man verschiedene multivariante Verfahren der Datenanalyse für die Ermittlung von Gruppen (Cluster) zusammengehöriger Objekte aus einer Grundmenge von numerisch beschriebenen Objekten[...](Wikipedia 2005)

Clustern *sing. Cluster <engl.> der*; -s, -[s]: 1. als einheitliches Ganzes zu betrachtende Menge von Einzelteilchen (Kernphysik)[...](Drosdowski u. a., 1997)

Communitites Eine Community (engl. community, Gemeinschaft) ist eine Gruppe von Personen, die gemeinsames Wissen entwickeln, Erfahrungen teilen und dabei eine eigene Identität aufbauen. Communities profitieren von dem Grundsatz, dass alle Teilnehmer zum Erfolg beitragen, indem sie ihr Wissen einbringen[...](Wikipedia 2005)

Harmonie *<gr.-lat.;* 'Fügung'>[...]2.ausgewogenes, ausgeglichenes Verhältnis der Teile zueinander[...](Drosdowski u. a., 1997)

IT Informationstechnik, Oberbegriff für die Informations- und Datenverarbeitung.(Wikipedia 2005)

IT Informationstechnologie

KI Künstliche Intelligenz (beziehungsweise AI vom englischen Artificial Intelligence) ist eine Fachdisziplin der Informatik mit interdisziplinärem Charakter. Ziel der KI ist es, Maschinen zu entwickeln, die sich verhalten, als verfügten sie über Intelligenz.(Wikipedia 2005)

Middleware [...] bezeichnet in der Informatik anwendungsunabhängige Technologien, die Dienstleistungen zur Vermittlung zwischen Anwendungen anbieten, so dass die Komplexität der zu Grunde liegenden Applikationen und Infrastruktur verborgen wird[...] (Wikipedia 2005)

Outsourcing Mit Outsourcing (dt. Auslagerung) wird in der Ökonomie die zeitlich begrenzte Abgabe von Unternehmensaufgaben und -strukturen an Drittunternehmen bezeichnet. Outsourcing ist somit eine spezielle Form des Fremdbezugs von bisher intern erbrachter Leistung, wobei die Dauer wie der Gegenstand der Leistung vertraglich fixiert werden. Dadurch wird Outsourcing von sonstigen Partnerschaften abgegrenzt[...] (Wikipedia 2005)

PDA Personal Digital Assistant (englisch für persönlicher digitaler Assistent) ist ein kleiner tragbarer Computer, der meist mit einem schnell startenden Betriebssystem ausgestattet ist und neben vielen anderen Programmen hauptsächlich für die persönliche Kalender-, Adress- und Aufgabenverwaltung benutzt wird.(Wikipedia 2005)

PIM Personal Information Management ist ein Anwendungsbereich für Software. Die Aufgabe entsprechender Software-Produkte besteht darin, Menschen bei der Verwaltung ihrer persönlichen Daten zu unterstützen. Dazu gehören typischerweise Kontaktdaten, Termine, Aufgaben, Notizen, und im erweiterten Verständnis auch Dokumente wie Briefe, Faxe und E-Mails. Personal Information Management ist die Voraussetzung der Vernetzung mit Groupware. Die bekannteste Software für diese Aufgabe ist Microsoft Outlook. Dabei machen die im Vergleich zu Outlook Express zusätzlich vorhandenen Merkmale die PIM-Funktionalität aus. PIM-SystemePIM werden oft auf PDAsPDA betrieben und/oder mit ihnen abgeglichen (Wikipedia 2005).

Profil *<lat.-it.(-fr)> das;* -s, -e: 1. Seitenansicht [eines Gesichts]; Umriss.[...](Drosdowski u. a., 1997)

Profiler [...] Ein Profiler erstellt Täterprofile. Die Tätigkeit bezeichnet man als profiling oder operative Fallanalyse. Dabei erstellt der Profiler ein charakteristischen Erscheinungs- und Persönlichkeitsbild eines unbekannten Straftäters anhand von Indizien, Spuren am Tatort und den Umständen der Straftat[...] (Wikipedia 2005).

prototypisch *von Prototyp (griech., 'der erste')* ein für die jeweiligen Zwecke voll funktionsfähiges Versuchsmodell eines geplanten Produktes oder Bauteils (Wikipedia 2005).

SmartPhones (schlaues Telefon), auch Mobile Digital Assistant (MDA), vereint den Leistungsumfang eines PDAs mit einem Mobiltelefon, wobei der Ansatzpunkt je nach Hersteller mehr der PDA oder das Mobiltelefon ist (Wikipedia 2005).

Soziale Netzwerke Soziale Netzwerke sind Netzwerke, die (1) in der Soziologie gegebene Interaktionsgeflechte (zum Beispiel Bekanntschafts-Netzwerke) abbilden, (2) in (zumeist) der Betriebswirtschaftslehre zielbezogene Organisationen (zum Beispiel informelle Zusammenschlüsse, Verbände) von Menschen umschreiben, die durch das Netzwerk einen Vorteil erfahren oder sich erhoffen[...](Wikipedia 2005)

Weblogs Ein Weblog (ein Kunstwort aus 'Web' und 'Logbuch'), üblicherweise einfach nur Blog genannt, ist eine Webseite, die periodisch neue Einträge enthält. Neue Einträge stehen an oberster Stelle, ältere folgen in umgekehrt chronologischer Reihenfolge[...](Wikipedia 2005)

Wiki Ein Wiki, auch WikiWiki oder WikiWeb genannt, ist eine im World Wide Web verfügbare Seitensammlung, die von den Benutzern nicht nur gelesen, sondern auch online geändert werden kann. Wikis ähneln damit Content Management Systemen[...](Wikipedia 2005)

Index

F

G

H

I

J

K

U

V

W

Z

Versicherung über Selbstständigkeit

Hiermit versichere ich, dass ich die vorliegende Arbeit im Sinne der Prüfungsordnung nach §24(5) ohne fremde Hilfe selbstständig verfasst und nur die angegebenen Hilfsmittel benutzt habe.

Hamburg, 3. August 2005	
Ort, Datum	Unterschrift