Agentenorientiertes Design für vernetzte Märkte

Klaus Dorer, Kurt Kammerer, Clemens Fritschi

living systems AG
Humboldtstr. 11, D-78166 Donaueschingen
klaus.dorer@living-systems.de

Abstract

Dieser Artikel beschreibt Design und Umsetzung eines Agentensystems für den Handel in vernetzten Märkten am Beispiel der Umgebung der Trading Agent Competition (TAC). Die Trading Agent Competition stellt eine Umgebung zur Verfügung, in der die Agenten als Reiseagenten tätig werden und in Konkurrenz mit anderen Agenten für ihre Kunden Hotels, Flüge und Unterhaltungsticket in multiplen, parallel stattfindenden Auktionen buchen. Die beschriebene Lösung stellt sich nicht nur als effizient bezüglich der Entwicklungszeit, sondern auch bezüglich der Laufzeitperformanz heraus und ging als Sieger aus dem zuletzt durchgeführten jährlichen internationalen TAC-Wettbewerb in Florida hervor.

Schlüsselworte

Multiagenten System, agentenorientiertes Softwareengineering, elektronische Märkte, Trading Agent Competition

1 Einleitung

Die erzielbare Wertschöpfung auf Märkten hängt auch im elektronischen Handel direkt von Marktliquidität und Transaktionskosten ab. Die Marktliquidität kann dabei gesteigert werden, indem sich ergänzende Märkte vernetzt werden.

Dabei steigt aber auch die Komplexität von elektronischem Handel, da z.B. der Kauf einer Ware und der Kauf der Transportkapazität gleichzeitig auf verschiedenen Marktplätzen stattfinden muss.

An der Universität von Michigan wurde in den vergangenen 2 Jahren eine Software Umgebung zur Simulation solch vernetzter Marktplätze erstellt. Dabei handelt es sich um eine Umgebung aus dem Touristik Bereich, die vernetzte Marktplätze für Flugtickets, Hotels und Freizeit Tickets simuliert. Sie erleichtert einerseits die technische Integrations gegenüber echten Marktplätzen und erlaubt so sich auf die wirtschaftlichen Probleme zu konzentrieren, andererseits dient sie auch dem Vergleich verschiedener Algorithmen zur Optimierung des Gewinns beim Handel in solchen Märkten. Dazu wird seit 2 Jahren jährlich ein internationaler Wettbewerb, die Trading Agent Competition (TAC) durchgeführt, in dem sich Mannschaften im Handel auf vernetzten Marktplätzen messen können (Wellman et al. 2001).

In diesem Papier zeigen wir, wie mit Hilfe von agentenorientiertem Design solche komplexen Umgebungen besonders effizient und intuitiv modelliert werden können. Dazu wird zunächst die TAC-Umgebung näher beschreiben. Danach wird das Design und die Strategie unseres Agentensystems erläutert, das im Oktober 2001 an der 2. TAC teilnahm. Schließlich werden die Ergebnisse der 2. TAC vorgestellt und diskutiert.

2 Die TAC-Umgebung

In der TAC-Umgebung haben autonome Agenten die Aufgabe, eine Reise für ihre (simulierten) Klienten zu organisieren, indem sie Flug, Hotel und Unterhaltungstickets in simultanen Auktionen kaufen. In jeder Auktionsrunde konkurrieren 8 Mannschaften darum, eine optimale Reise für ihre jeweils 8 Kunden zu organisieren. Ein Reisepaket besteht aus dem Flug nach Tampa, bis zu 4 Nächten Unterkunft in einem Hotel, optional zu ersteigernden Unterhaltungstickets und dem Rückflug. Die Präferenzen der Kunden

werden zufällig durch den Auktionsserver am Anfang jeder Auktionsrunde zugewiesen. Sie beinhalten Zeitpunkt und Dauer des Aufenthalts, wie viel der Kunde bereit ist, für das Luxushotel zusätzlich zu zahlen und welche Unterhaltung der Klient bevorzugt.

Die Tickets werden in 28 parallel stattfindenden Auktionen in 3 unterschiedlichen Auktionsarten verkauft.

- Hotel: Es gibt 2 Arten von Hotels: Das Tampa Towers (Luxushotel) und das Shoreline Shanty (Economy). Die Kunden wünschen einen Aufenthalt zwischen 1 und 4 Nächten und haben jeweils Präferenzen für Luxus- oder Economy Zimmer. Jedes Hotel hat 16 Räume und jede Übernachtung muss in einer eigenen Auktion ersteigert werden. Dabei muss für einen Kunden jede Nacht im selben Hotel gebucht werden. Kann für eine Nacht kein Zimmer ersteigert werden, storniert der Kunde die komplette Reise. Die Auktionen sind englische Auktionen zum 16. Preis. D.h. die höchsten 16 Angebote in einer Auktion erhalten den Zuschlag zum Preis des 16. höchsten Angebots. Hotelräume können nicht wiederverkauft werden und Angebote können nicht zurückgenommen werden. Die Dauer von Hotelauktionen variieren zufällig bei einer Mindestdauer von 5 Minuten.
- Flug: Es gibt 8 Auktionen für Flüge: 4 für Hinflüge an den Tagen 1 bis 4 und 4 für Rückflüge an den Tagen 2 bis 5. Flugtickets sind immer verfügbar, ein Angebot zum Ask-Preis oder darüber wird also immer erfolgreich sein. Im Unterschied zur ersten TAC stieg der Preis für einen Flug aber mit zunehmender Auktionsdauer immer stärker an. Dadurch wurden frühe Entscheidungen für Flüge begünstigt, allerdings mit der Gefahr, nicht alle gewünschten Übernachtungen ersteigern zu können.
- Unterhaltungstickets: Es gibt 3 Arten von Unterhaltungstickets, die an jedem der ersten 4 Tage vorhanden sind: Karten für ein Museum, einen Freizeitpark und Krokodilwrestling. Die teilnehmenden Mannschaften werden am Anfang zufällig mit einem Set solcher Karten ausgestattet. Die Karten können dann während einer Auktionsrunde in einer continuous double auction verkauft und gekauft werden, d.h. immer wenn ein Angebot zum Verkaufspreis einer anderen Mannschaft gemacht wird, kommt ein Vertragsabschluß zustande.

Für die Berechnung des Ergebnisses werden sowohl Kundenzufriedenheit als auch die eigenen Kosten berücksichtigt. Im Detail berechnet sich das Ergebnis für einen Kunden wie folgt:

Ergebnis = Kundenzufriedenheit - Kosten, wobei
Kundenzufriedenheit = 1000 – Reisestrafe + Hotelbonus + Freizeitbonus.

Die Reisestrafe beträgt 100 Punkte für jeden Tag, den der Kunde früher oder später anreisen bzw. abreisen muss. Der Hotelbonus wird zugewiesen, wenn der Klient im Luxushotel untergebracht wird und entspricht dem Betrag, den der Kunde bereit war zusätzlich für das Luxushotel zu zahlen. Der Freizeitbonus wird für jede vom Kunden gewünschte Unterhaltungskarte erteilt, die gekauft wurde. Die Höhe entspricht dem präferierten Kaufpreis des Kunden. Das Gesamtergebnis ist die Summe der Ergebnisse aller 8 Kunden. Ziel ist es, das Gesamtergebnis zu optimieren.

Die Herausforderung besteht darin, in den 28 simultanen Auktionen ein optimales Paket an Tickets zu ersteigern, wobei die Resultate der Auktionen von einander abhängen. Ein früh gekauftes, preiswertes Flugticket z.B. könnte sich als kontraproduktiv erweisen, wenn später in der Hotel Auktion die Preise für die erste Nacht des Aufenthalts explodieren. Ein Flug am folgenden Tag würde zwar die Kundenzufriedenheit beeinträchtigen, könnte aber das Gesamtergebnis positiv beeinflussen. Um die Ergebnisse über eine längere Zeit mitteln zu können, werden mehrere solcher Auktionsrunden durchgeführt. Eine ausführlichere Beschreibungen der TAC-Umgebung und der Auktionsmechanismen findet man bei (Wellman et al. 2001; Stone et al. 2001; Greenwald u. Stone 2001) oder auf der TAC-homepage unter http://auction2.eecs.umich.edu.

3 Agentenorientiertes Design

Um der zunehmenden Komplexität von Umgebungen gerecht zu werden, wurden Methoden für Analyse und Software Design auf immer höheren Abstraktionsebenen entwickelt (z.B. prozedurale Programmierung, objektorientierte Programmierung, Design Patterns). Es kann argumentiert werden (Jennings u. Wooldridge 2001; Weiß 2001), dass agentenorientiertes Software Engineering die nächste Abstraktionsebene darstellt. Neue Methoden wurden entwickelt, um Analyse und Design von agentenbasierten Systemen zu unterstützen wie zum Beispiel KGR (Kinny et al. 1996), Agent-UML (Bauer et al. 2000) oder Gaia (Wooldridge et al. 2000).

Gaia beispielsweise definiert 2 Modelle für die Analyse und 3 Modelle für das Design eines Agentensystems. In der Analyse wird ein Rollenmodell und ein Interaktionsmodell der Agenten entworfen. In der Design Phase wird aus dem Rollenmodell ein Agentenmodell mit darin enthaltenen Agententypen entworfen, ein Service Modell mit den Services, die die Agenten zur Verfügung stellen, sowie ein Acquaintance Modell, das die Kommunikationsverbindungen zwischen den Agenten darstellt (Wooldridge et al. 2000).

3.1 Living agents Design

Das living agents TAC Team basiert auf dem living agents runtime system der living systems AG. Living agents ist Basis für mehr als 30 Internetplattformen in den Bereichen Marktplätze, B2B, Logistik und Finanzwesen. Während die Agententechnologie selbst die Grundlage für eine solch breite Anwendbarkeit liefert, sorgen agentenorientierte Softwaremethoden und Tools für eine strukturierte und effiziente Modellierung komplexer Umgebungen.

Als Design Umgebung wurde die living markets development suite (LMDS) verwendet, der eine mit Gaia vergleichbare Methodik zu Grunde liegt. Sie erlaubt die Definition eines Agenten Szenarios, das im wesentlichen eine kompakte Repräsentation der 3 Modelle für die Design Phase von Gaia darstellt. Im folgenden beschreiben wir die Schritte, die notwendig sind für das Design eines Agentensystems:

1. Definition der Agenten Typen
2. Definition der Interaktionen zwischen den Agenten

3.1.1 Agenten Typen

Zunächst müssen die Agenten Typen definiert werden. Als Ausgangspunkt dient dabei in der Regel eine Organisation, um mit Hilfe von Menschen die Aufgabe optimal zu lösen. Beim Beispiel TAC, bei dem Kundenzufriedenheit entscheidend ist, liegt es auf der Hand, dass eine optimale Kundenbetreuung dadurch gewährleistet werden kann, dass für jeden Kunden ein Betreuer zuständig ist. Dementsprechend wurde der Agententyp ‚Kundenbetreuer' eingeführt und 8 Instanzen dieses Typs später auf dem Agentensystem instanziiert, für jeden Kunden einer.

Ein Agenten Typ besteht aus einem Namen, einer Beschreibung, einem Business Logik Typ, den Verantwortlichkeiten und Befugnissen. Der Business Logik Typ gibt an, wie das Business Wissen im Agenten repräsentiert und verarbeitet wird. Derzeit werden 3 Repräsentationen unterstützt: Workflows, Entscheidungsbäume und Verhaltensnetzwerke. Alle drei erlauben die Spezifikation von Business Wissen in einfacher Form oberhalb der Abstraktionsebene von Programmiersprachen. Dadurch kann das Wissen auch von nicht-Informatikern erstellt und angepasst werden. Gleichzeitig kann dadurch auch das Verhalten der Agenten zur Laufzeit modifiziert werden, ohne die Agentenplattform neu starten zu müssen. Die Spezifikation des Business Wissens und der Verantwortlichkeiten für die TAC-Umgebung wurde als Workflows umgesetzt und wird in Kapitel 3.2.1 beschrieben. Ein Rechte Modell wurde in dieser Umgebung nicht benötigt.

Insgesamt wurden 6 Agenten Typen definiert (in Klammern stehen die Anzahl der Agenten Instanzen dieses Typs):

- TACClient (8): Kundenbetreuer, verantwortlich für die optimale Zusammenstellung der Tickets für den zugewiesenen Kunden
- TACAuctioneer (4): Auktionär, verantworltich für das Bieten in den zugewiesenen Auktionen auf Weisung der Kundenbetreuer
- TACEntertainmentAuctioneer (1): Freizeit Ticket Auktionär, spezialisiert für das Handeln in den continuous double auctions der Freizeit Tickets
- TACDataGrabber (5): verantwortlich für das Einholen der aktuellen Auktionsinformationen
- TACManager (1): koordiniert die Agenten zu Beginn und am Ende von Auktionen
- TACResultGrabber (1): verantwortlich für die Verwaltung der Ergebnisse früherer Auktionen

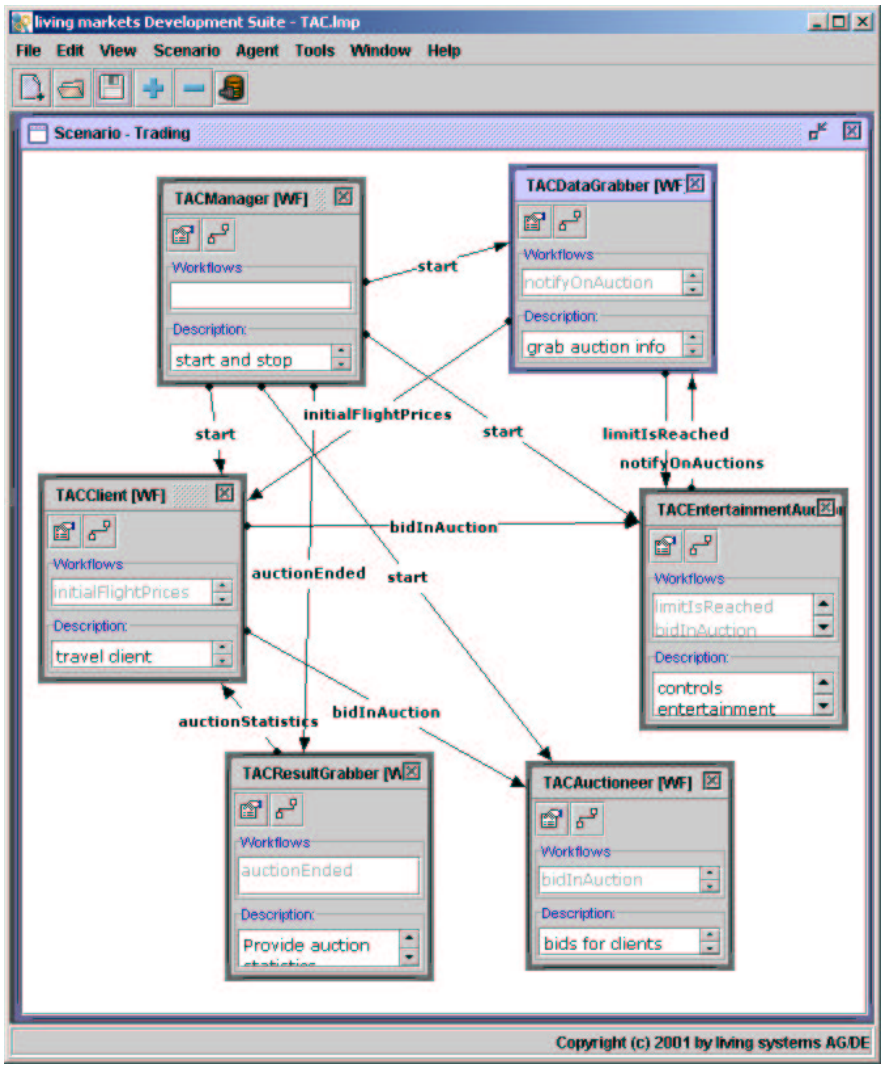

Abb. 1. TAC Agenten Szenario in der LMDS mit den Agenten Typen und Interaktionen.

3.1.2 Interaktionen

Neben den Agenten Typen definiert ein Agenten Szenario außerdem die Interaktionen zwischen den Agenten. Eine Interaktion ist definiert durch den Sender einer Nachricht, den Empfänger und einem Nachrichtennamen. Auf Empfängerseite definiert eine Interaktion direkt einen Service, den der Empfänger beim Empfang der entsprechenden Nachricht zur Verfügung stellt.

Im Beispiel in Abbildung 1 ist spezifiziert, dass der TACClient dem TACAuctioneer die Nachricht ‚bidInAuction' schicken kann. Ein Agenten Szenario kann so Aufschluss darüber geben, an welchen Stellen z.B. Flaschenhälse für die Kommunikation auftreten könnten.

Eine Interaktion definiert nicht, wann eine Nachricht gesendet wird. Dies ist Teil des Business Wissens, das im folgenden beschrieben wird.

3.2 Implementierung

Für die Implementierung eines Agentensystems ist es zum einen nötig, das Verhalten des Agenten zur Laufzeit zu definieren und zum anderen muss der Agent in die Umgebung eingebettet werden (grounding). Für den im Beispiel gewählten Business Logik Typ ‚Workflow' bedeutet das:

1. Definition der Services eines Agenten
2. Definition der Wahrnehmungen und Aktionen der Agenten

3.2.1 Services

Nachdem mittels der Interaktionen bereits die Services jedes Agenten deklariert wurden, müssen Sie jetzt noch im einzelnen mit Business Wissen versehen werden. Ein Service besteht aus seinem Namen und Typ, der Ein- und Ausgabeinformation sowie dem Inhalt. Ein Service wird aufgerufen, wenn der Agent eine Nachricht mit entsprechendem Namen erhält. Der Typ eines Service ist bereits durch die Wahl des Business Logik Typs festgelegt und ist in unserem Beispiel vom Typ Workflow. Die Grundbausteine von Workflows, sowie alle anderen Business Logik Typen, sind Aktionen und Wahrnehmungen des Agenten die so genannten Capabilities. Die Abfolge der Aktionen und Wahrnehmungen werden bei einem Workflow als Ablaufplan definiert. Als Beispiel für einen Workflow in der TAC-Umgebung wurde der ‚start' Service des TACDataGrabber dargestellt (Abbildung 2). Dieser Service signalisiert dem TACDataGrabber, dass er nun beginnen kann, Auktionsinformationen einzuholen und entsprechend der aktuellen Gebote den TACEntertainmentAuctioneer (TEA) informieren soll.

3.2.2 Wahrnehmungen und Aktionen

Nachdem der Ablauf eines Services definiert ist, müssen die noch nicht vorhandenen Aktionen und Wahrnehmungen definiert und implementiert werden. Wahrnehmungen liefern dem Agenten Information über seine Umgebung. Aktionen erlauben dem Agenten die Umgebung aktiv zu beeinflussen. Beide werden als Java Methoden implementiert und in so genannten Capability Klassen zusammengefasst. Diese Capabilities können von beliebigen Business Logiken verwendet und wiederverwendet werden.

Im Beispiel Service ‚start' des TACDataGrabber in Abbildung 2 wurden 6 Wahrnehmungen und 3 Aktionen definiert:

Wahrnehmungen

- moreAuctionsResponsibleFor: wahr, solange es offene Auktionen gibt, für die der Agent verantwortlich ist.
- auctionEnded: wahr, wenn die Auktion im aktuellen Kontext beendet ist.
- existingNotifications: wahr, wenn weitere Kauf- oder Verkaufsanfragen des TEA für eine Auktion vorhanden sind.
- notifyMeOnPriceAboveLimit: wahr, wenn der TEA ein Ticket verkaufen will.
- actualPriceAboveLimit: wahr, wenn die vom TEA spezifizierte Preisgrenze erreicht oder überschritten ist.
- actualPriceBelowLimit: wahr, wenn die vom TEA spezifizierte Preisgrenze nicht erreicht ist.

Aktionen

- getAuctionPriceFromServer: holt den aktuellen Stand der Auktion vom TAC server.
- send: schickt einem anderen Agenten eine Nachricht. Die Eingabe Parameter dieser Aktion sind der Name der Nachricht und des Empfängers, sowie der Inhalt der Nachricht.
- deleteNotifyEntry: löscht die aktuelle Anfrage des TAE

Das so definierte Agenten Szenario kann dann in XML repräsentiert und in Dateien abgelegt werden. Diese XML Repräsentation dient dann dem living agents Agenten Server als Konfiguration für die einzelnen Agenten.

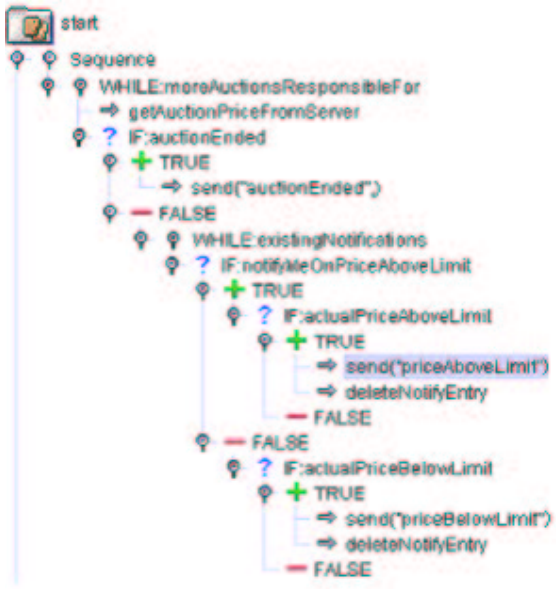

Abb. 2. Definition eines Workflows am Beispiel des Service ‚start' des TACDataGrabber.

3.3 Strategie

In diesem Abschnitt gehen wir auf die Strategie der Agenten bei der Teilnahme in der Trading Agent Competition ein. Der Strategie lagen folgende 2 Annahmen zugrunde, die sich im Verlauf des Wettkampfs bestätigten:

1. Durch die ständig steigenden Flugpreise sind frühe Entscheidungen für Flugtickets wichtig.
2. Insbesondere gute Mannschaften versuchen ihren eigenen Nutzen zu maximieren. Sie nehmen also nicht das Risiko in Kauf, den Nutzen des Gegners durch Kampfgebote zu reduzieren, was in den 16. Preis Auktionen der Hotels ohnehin schwierig ist.

Während bei der 1. Trading Agent Competition, bedingt durch andere Auktionsregeln bei Flugzeugtickets, die Strategie des Siegerteams ATTac (Stone et al. 2001) vorsah, Entscheidungen so spät wie möglich im Verlauf der Auktionen zu treffen, fordern obige Annahmen bei den Auktionsregeln der 2. Trading Agent Competition eine möglichst frühe Entscheidung von den Agenten, um Geld für Flugtickets zu sparen. Dementsprechend wurden Flugtickets früh gekauft und entsprechend hoch in Hotelauktionen geboten. Lediglich Freizeit Tickets wurden während des gesamten Verlaufs einer Auktionsrunde gehandelt. Während diese Strategie in den Vorrunden weniger effektiv war, da schwächere Mannschaften die Hotelpreise explodieren ließen, war sie in Halbfinale und Finale der Garant für die höchsten Ergebnisse (siehe Tabelle 1).

Die Strategie im Detail:

1. Der TACManager wartet auf den Start einer neuen Auktionsrunde, fordert dann die Kundeninformationen an und informiert alle anderen Agenten über den Start der Auktion.
2. Die TACDataGrabber für die Flugtickets fordern die aktuellen Preise an und senden sie an die TACClients (Kundenbetreuer).
3. Die 8 Kundenbetreuer berechnen das für ihren Kunden optimale Reisepaket basierend auf den Kundenwünschen, den aktuellen Flugpreisen und den aus früheren Auktionen zu erwartenden Hotelpreisen (siehe unten). Danach schicken sie die Ticketwünsche sowie die maximalen Preise der Freizeit Tickets an die TACAuctioneers und TACEntertainmentAuctioneers.
4. Die TACAuctioneers bieten für die gewünschten Hotels und Flugtickets.
5. Der TACEntertainmentAuctioneer informiert den TACDataGrabber für Freizeit Ticket Preise über die Auktionen in denen er kaufen oder verkaufen möchte. Sobald er vom TACDataGrabber über ein günstiges Gebot informiert wird, kauft bzw. verkauft er ein Ticket. Je länger die Auktion dauert, desto weniger pofitable Gebote akzeptiert der TEA, bis am Ende der Auktion auch Preise nahe am Profitlimit akzeptiert werden.

Das unter optimalen Bedingungen (alle gewünschten Tickets können ersteigert werden) bestmögliche Reisepaket wurde berechnet mittels vollständiger Permutation aller möglichen Reisepakete:

```
for (Ankunftstag = 1 to 4)
for(Abflugstag = Ankunftstag + 1 to 5)
  for (beide Hotel Typen)
     benefit = 1000     - Preis für Hinflug
                        - Preis für Rückflug
                        - sum(mittlerer Hotelpreis)
                        – Reise Strafe
                        + Hotel Bonus
                        + Freizeit Ticket Bonus
```

Dabei sind sowohl die Preise für Hin- und Rückflug als auch die Reise Strafe und der Hotel Bonus bekannt. Die mittleren Hotelpreise wurden genau wie die Preise für Freizeit Tickets basierend auf den entsprechenden Preisen früherer Auktionen geschätzt. Die Agenten passten sich so dem Verlauf der Auktionen mit der Zeit immer besser an.

4 Ergebnisse

Die 2. Trading Agent Competition wurde in insgesamt 4 Runden durchgeführt: einer Qualifikationsrunde, einer Zwischenrunde für die Bestimmung der Gruppen für das Halbfinale, dem Halbfinale und Finale. Insgesamt nahmen die Agenten an über 1800 Auktionsrunden mit mehr als 50000 Auktionen teil.

Tabelle 1. Spielplan der 4 Runden der 2. Trading Agent Competition

Runde	Teams	Auktionsrunden	Datum
Qualifikation	28	992	10.9. – 17.9.
Zwischenrunde	19	775	24.9. – 5.10.
Halbfinale	16	12	14.10.2001
Finale	8	24	14.10.2001

Das Finale wurde über 24 Auktionsrunden ausgetragen, wobei alle 8 Finalteilnehmer in denselben Auktionen bieten mussten. Tabelle 2 zeigt die durchschnittliche Punktzahl der Finalteilnehmer sowie die Standardabweichung. Unser Team aus 19 Agenten konnte sich am Ende mit durchschnittlich 3670 Punkten vor dem Vorjahressieger ATTac durchsetzen.

Tabelle 2. Ergebnisse der Finalrunde der 2. Trading Agent Competition 2001 in Tampa, Florida.

Rang	Team	Punkte ∅	Std. Abw.	Institution
1	Livingagents	3670	622	living systems AG, GER
2	ATTac	3622	692	AT&T Labs – Research, USA
3	whitebear	3513	700	Cornell University, USA
4	Urlaub01	3421	698	Pennsylvania State Univ., USA
5	Retsina	3352	668	Carnegie Mellon University, USA
6	SouthamptonTAC	3254	1467	University of Southampton, UK
7	CaiserSose	3074	656	University of Essex, UK
8	TacsMan	2859	1054	Stanford University, USA

5 Zusammenfassung

Beim TAC-Wettbewerb geht es nicht um die Maximierung eines individuellen Kundennutzens, sondern um die Maximierung des Gesamtnutzens einer ganzen Gruppe von Kunden. Damit ist nicht die isolierte Optimierung ausschlaggebend für den Erfolg, sondern die geschickte Allokation eines Travelservices im betreuten Kundennetzwerk der Gestalt, dass der höchstmögliche Nutzenbeitrag für das Kundennetzwerk erzielt werden kann. Diese Zuordnungsaufgabe gestaltet sich anspruchsvoll durch die Dynamik des

Travelmarktes in Form volatiler Preise und Verfügbarkeiten und die Trade-Off-Betrachtungen zwischen Kunden. Aus Sicht der Gesamtaufgabe macht es etwa Sinn, Hotels zwischen zwei Kunden zu tauschen, wenn der Nettonutzenbeitrag positiv ist. Durch die Anzahl zu betrachtender Knoten in diesem Wertschöpfungsnetzwerk sowie die Dynamik des Marktumfelds ist ein zentraler Lösungsansatz aus Komplexitätsgründen zum Scheitern verurteilt. Hingegen erweist sich ein dezentraler Ansatz, bei dem Kunden einzeln durch elektronische Kundenbetreuer repräsentiert sind, als robust. Diese sorgen durch geschickte Interaktion dafür, dass der Gesamtnutzen des Wertschöpfungsnetzwerkes optimiert wird.

Was sich hier als relativ exotisches Travel-Beispiel präsentiert, hat ökonomische Relevanz weit über diesen Bereich hinaus. Jede Wertschöpfung in einem Business-Netzwerk, egal ob Supply-, Distributions- oder Produktionsnetzwerk, soll den höchstmöglichen Gesamtnutzen erzielen. Auch für diese Beispiele trifft wie im Travel-Fall die eingeschränkte Planbarkeit zu, weil sich Umweltbedingungen unvorhersehbar ändern können. Der Ansatz, Einzelinstanzen eines Netzwerks durch Software-Agenten zu repräsentieren, ist auch hier nahe liegend und kann wie im Travel-Fall die Gesamtwertschöpfung steigern.

Literaturverzeichnis

Bauer B, Müller JP, Odell J (2001) Agent UML: A Formalism for Specifying Multiagent Software Systems. Int. Journal of Software Engineering and Knowledge Engineering 11(3) S. 207-230.

Greenwald A, Stone P (2001) Autonomous Bidding Agents in the Trading Agent Competition. IEEE Internet Computing, March/April, S. 52-60.

Jennings NR, Wooldridge M (2001) Agent-Oriented Software Engineering. In: Bradshaw J (Hrsg) Handbook of Agent Technology, AAAI/MIT Press.

Kinny D, Georgeff M, Rao A (1996) A Methodology and Modelling Technique for Systems of BDI Agents. Proceedings of the 7th European Workshop on Modelling Autonomous Agents in a Multi-Agent World. Springer, Berlin.

Stone P, Littman L, Singh S, Kearns M (2001): ATTac-2000: An Adaptive Autonomous Bidding Agent. In: Proceedings of the 5th International Conference on Autonomous Agents.

Weiß G (2001) Agentenorientiertes Software Engineering. Informatik Spektrum 24 (2), S. 98-101.

Wellman MP, Wurman PR, O'Malley K, Bangera R, Lin S, Reeves D, Walsh WE (2001) Designing the Market Game for a Trading Agent Competition. IEEE Internet Computing, March/April, S. 43-51.

Wooldridge M, Jennings NR, Kinny D (2000) The Gaia Methodology for Agent-Oriented Analysis and Design. Journal of Autonomous Agents and Multi-Agent Systems 3 (3), S. 285-312.