Kommunikation in Multi-Agenten-Systemen

PS Einführung in die Bioinformatik (SS 99)

Markus Amersdorfer
markus.amersdorfer@sbg.ac.at

Michael Noisternig
michael.noisternig@sbg.ac.at

Version: 23. März 1999
Uhrzeit: 12:30 - 13:00
Gruppe: 5

LV-Leiter: Dipl.-Ing. Mag. Roland Schwaiger
rschwaig@cosy.sbg.ac.at

Zusammenfassung

Dieser Artikel gibt eine Übersicht zum Thema Agenten, Mulit-Agenten-System (MAS) und Kommunikation in MAS.

Inhaltsverzeichnis

Kapitel 1

Einleitung

1.1 Was ist ein Agent?

Zu allererst muß gesagt werden, daß es keine allgemein akzeptierte Definition eines Agenten gibt. Das Online-Wörterbuch "Merriam-Webster" [1] definiert einen Agenten folgendermaßen:

> "Ein Instrument, durch welches eine führende Intelligenz Resultate erreicht."
> bzw. "Jemand, der von einer anderen Person authorisiert ist, anstelle dieser zu handeln als: [...] c) ein (geschäftlicher) Repräsentat [...]"

Im "The Webster Dictionary" [2] wird der Begriff definiert als:

> "Ein Agent ist eine Komponente, welche für oder anstelle einer anderen autoritätsbesitzenden Komponente arbeitet."

Diese Definitionen sind allerdings schwer verständlich.
In [3] beschreiben M. Genesereth und S. Ketchpel einen Softwareagenten als ein Programm, das über eine ausdrucksvolle Agentenkommunikationssprache (ACL) mit anderen Programmen kommunzieren kann, um Informationen (Daten und Code) auszutauschen. Näheres dazu siehe 2.2.
Agenten haben eine nachrichtenbasierte Schnittstelle (siehe 2.1), die unabhängig ist von internen Datenstrukturen und Algorithmen (design autonomy). Das heißt, die programmiertechnische Implementierung spezieller Softwareagenten ist frei wählbar, da die Schnittstellen zur Außenwelt für die Kommunikation standardisiert sind.
Auf Nachrichten zeigen Agenten ein bestimmtes Verhalten. Jedoch besitzen auch Objekte in der Objektorientierten Programmierung (OOP) diese Eigenschaften.

Bemerkung: Bei Verwendung der Bezeichnung "Softwareagent" stellt sich folgende Frage: "Was ist ein Hardwareagent?" Eine Antwort dazu ist schwer zu finden, jedoch kann man diesen wohl am ehesten als Robotor, sprich Softwareagent mit Hardware, definieren.

1.2 Was ist der Unterschied zwischen einem Objekt in der OOP und einem Agenten?

Ein Unterschied ist, daß Agenten eine einheitliche und immer gleich verständliche Sprache haben. Dies ist jedoch nicht ausreichend, da es zum einem, wie später beschrieben wird, Nachrichten gibt, die gar nicht klar definiert sind und somit immer anders verstanden werden können, und zum anderen, weil auch Programme in einem System, in dem ein Kommunikationsprotokoll als Sprache verwendet wird, diese Eigenschaften erfüllen, z.B. Mail-Server mit SMTP als Kommunikationssprache.
M. P. Singh [4] liefert eine wesentlich bessere Definition eines Agenten:

> "Ein Agent ist ein Programm, das auf seine Umgebung reagieren und sowohl mit anderen Agenten zusammenarbeiten (interoperability) als auch ohne sie agieren kann. Agenten können selbstständig handeln und insofern Handlungen ablehnen oder annehmen (execution autonomy)."

Auf den Punkt bringt es [5]:

> "Im Gegensatz zu Objekten haben Agenten Ziele und handeln aktiv, um diese zu erreichen."

1.3 Was sind also die wichtigsten Eigenschaften eines Agenten?

- *Autonomie:* ein Agent kann beliebig aufgebaut sein (design autonomy) und er kann selbst wählen, welche Handlungen er ausführt (execution autonomy)
- *Reaktivität:* jeder Agent muß auf Veränderungen in seiner Umgebung entsprechend reagieren können; das bedeutet nicht unbedingt eine nach außen sofort erkennbare Reaktion, sondern möglicherweise auch einfach nur eine agenteninterne Veränderung von Zuständen oder Zielen
- *Interoperabilität:* Agenten müssen Informationen mit anderen Komponenten (Agenten, Programmen, Menschen) austauschen können

1.4 Klassifizierung von Agenten

Weitere Eigenschaften kann man auch zur Klassifizierung verwenden [6], [5], [7]:

- *Intelligenz*: ein Agent kann die beste Lösung in einer bestimmten Situation finden
- *Adaptivität, Flexibilität:* ein Agent verändert sein Verhalten aufgrund seiner früheren Erfahrungen
- *Zielorientierung:* ein Agent arbeitet aktiv, um ein Ziel zu erreichen
- *Charakter:* ein Agent besitzt eine "Persönlichkeit" und einen emotionalen Status

- *Mobilität:* ein Softwareagent kann sich zwischen verschiedenen Rechnern bewegen

Den Charakter eines intelligenten Agenten bestimmt vor allem, ob der individuelle mentale Status des Agenten oder die sozialen Aspekte der Kommunikation betont werden. Nach M. P. Singh [4] wird ein mentaler Status am einfachsten durch "beliefs" und "intentions" beschrieben, die die gesamte Handlungsweise des Agenten bestimmen.

Von besonderer Bedeutung ist das BDI-Modell *(belief-desire-intention)*. Wie in [8] ersichtlich ist, wird hierbei der interne Status eines Agenten als ein Konstrukt von Wissen und daraus resultierenden Wünschen und Absichten dargestellt. Ein Agent hat hierbei Ziele bzw. Wünsche und sucht einen Weg, diese mit Hilfe seines Wissens zu erreichen. Aufgrund neu erlernten Wissens kann er dann entweder den Weg oder das Ziel ändern.

Ein Beispiel:

> Ich glaube, daß ich gut bezahlt werde, wenn ich hart arbeite.
> Ich wünsche mir einen Ferrari.
> Also arbeite ich hart, um mir einen Ferrari zu kaufen.
>
> Nach mehreren Jahren harter Arbeit habe ich noch immer keinen Ferrari!
> Also könnte ich
>
> - mir einen Lotterieschein kaufen (meine Absicht ändern)
> - erkennen, daß das nur ein Traum ist, und auf einen Mazda hinarbeiten
> - erkennen, daß harte Arbeit nicht genug Geld bringt und auf kriminelle Weise zu Geld kommen.

Soziales Verhalten ist das erklärte Ziel von Agentenentwicklern; ein Ziel, daß noch lange nicht erreicht ist. Solche *sozialen Agenten* sollten zum Beispiel alle Gesprächspartner gleich behandeln, den Kontext eines Gespräches verstehen und alle Kommunikationsarten beherrschen (siehe 2.2.2). [4]

1.5 Anwendungsgebiete für Agenten

Im folgenden einige Projekte der Software Agent Group [9] am MIT, gegründet von Pettie Maes.

1. *Butterfly* (1998-Present)
   by Neil W. Van Dyke
   Butterfly ist ein Agent, der stichprobenartig tausende laufende IRC-Channels (Internet Relay Chat) dursucht, und für den Benutzer interessante vorschlägt.

2. *Amalthea* (1996-Present)
   by Alexandros G. Moukas, Pattie Maes
   Anstatt einen Agenten mit der übermächtigen Aufgabe zu betrauen, für den Benutzer relevante jedoch weitverstreute Informationen des Internets zu sammeln, basiert Amalthea auf einem Multi-Agenten-System, welches diese Aufgabe mittels mehrerer Agenten ausführt und dem Benutzer das Ergebnis präsentiert. Weiters paßt sich das System den sich verändernden Bedürfnissen des Benutzers an.

3. *Yenta* (1995-Present)
   by Lenny Foner
   Yenta ist ein agentenbasiertes System, welches verschiedene Benutzer mit gleichen Interessen oder Arbeitsaufgaben gruppiert und zu zwischenmenschlicher Kommunikation anregt, dabei jedoch nicht die Privatshpäre der Einzelnen Gruppenmitglieder vernachlässigt.

4. *Challenger* (1996)
   by Alexandros G. Moukas, Anthony Chavez, Pattie Maes
   Challenger ist ein Multi-Agenten-System, welches für die Durchführung von Aufgaben die Rechenzeit verschiedener Computer verwendet.

5. *Webdoggie* (1994-1995)
   by Yezdi Lashkari, Pattie Maes
   Webdoggie ist ein System, welches mittels zusammenarbeitendem Filtern für den Benutzer möglicherweise interessante Internetseiten findet.

1.6 Multi-Agenten-Systeme (MAS)

Eine wichtige Errungenschaft der heutigen Informationstechnologie ist das Konzept der *Verteilten Künstlichen Intelligenz* (Distributed Artificial Intelligence = DAI). Wie K. C. Ranze und H. J. Müller in [10] beschreiben, dient dies der Lösung von immer größer werdenden und mehr Rechenzeit in Anspruch nehmenden Problemen. Ein wichtiger Vorteil der DAI gegenüber "normaler" Mulitprozessor-Aufgabenverteilung liegt darin, daß die einzelnen ausführenden Programme charakterisiert werden durch eigenständiges agieren in einer Welt, die entweder einen realen Weltausschnitt darstellt oder die künstlich mittels Objekten beschrieben ist. Diese als Agenten bezeichneten Programme kommunizieren nicht nur, um Services anzuforden und Ergebnisse abzuliefern, sondern vielmehr können selbständig Aufgaben angeboten, es kann verhandelt bzw. auch die Durchführung von Aufträgen verweigert werden.
Ein solches System von Agenten wird als *Mulit-Agenten-System* bezeichnet.

Kapitel 2

Kommunikation

2.1 Kommunikations-Mechanismen

Unter Kommunikations-Mechanismen in Zusammenhang mit Agenten und MAS wollen wir Methoden verstehen, um Nachrichten zwischen Agenten auszutauschen. Als die zwei wichtigsten Mechanismen sind zu nennen:

- *Message-Passing (Bringschuld):* Der benachrichtigende Agent A ist selbst dafür verantwortlich, daß seine Nachricht zu einem bestimmten Agenten B gelangt. Das heißt, er muß in einer Verbindung zu seinem Kommunikationspartner die Nachricht zu diesem “bringen”.
- *Blackboarding (Holschuld):* Dieser Mechanismus ist vergleichbar mit der Funktionsweise von Newsgroups. Agent A postiert seine Nachricht bei einer Zentrale, über die sämtliche Nachrichten laufen. Sein Kommunikationspartner Agent B ist nun selbst dafür veranwortlich, eigentständig die Nachricht bei der Zentrale abzuholen.

Diese Zentrale kann entweder implementiert werden als herkömmliche Datenbank, oder aber auch als ein eigener Agent C, der sich zur Verfügung stellt um die Kommunikation zwischen anderen zu ermöglichen. In diesem Fall allerdings ergibt sich zwischen den Agenten A und C wiederum eine auf Message-Passing basierende Kommunikation.

2.2 Ansätze für Kommunikation

Um Multi-Agenten-Systeme implementieren zu können, benötigt man als grundlegendes Werkzeug der Kommunikation eine Sprache. In [3] werden zwei übliche Ansätze für Sprachen genannt:

- Skriptsprachen
- deklarative Sprachen.

Skriptsprachen (wie TCL, Apple Events, Telescript) tauschen ganze Prozeduren aus und sind somit einfach und leistungsfähig. Deren Nachteil allerdings liegt darin, daß sie nur unidirektional und schwierig zu koordinieren sind, wenn mehrere Skripts von mehreren

Agenten gleichzeitig ankommen.
Deklarative Sprachen verstehen Kommunikation als Austausch von deklarativen Aussagen wie zum Beispiel Definitionen oder Annahmen. Das heißt, sie können ohne Interpreter ausgeführt werden. Wie T. Finin in [11] beschreibt, sind sie einerseits ausdrucksfähig genug, um vielfältige Informationen weiterzugeben und andererseits gleichzeitig kompakt.
Die Entwicklung von nachfolgenden deklarativen Sprachen wird vom ARPA Knowledge Sharing Effort vorangetrieben. Sie besitzen einen erweiterungsfähigen, englischsprachigen Wortschatz, um die Lesbarkeit für Menschen zu erhöhen.

2.2.1 KIF

KIF (Knowledge Interchange Format) ist eine Sprache mit Prädikatenlogik erster Ordnung und einigen Erweiterungen, um ihre Ausdruckskraft zu erhöhen. Laut M. Genesereth und S. Ketchpel [3] soll KIF jedoch in erster Linie einfache Daten logisch darstellen können und besitzt dazu logische Operatoren wie Negationen, Disjunktionen, Regeln usw. Weiters kann KIF Wissen von anderen Wissensbasen beziehen.
Weil KIF jedoch kontextunabhängig ist, müßte jede Nachricht jedoch implizite Informationen über den Empfänger, den Sender, die Zeit einer Nachricht usw. enthalten. Die Effizienz einer Kommunikation kann erhöht werden, indem sprachliche Mittel verwendet werden, bei denen der Kontext eine Rolle spielt.

2.2.2 KQML

KQML (Knowledge Query and Manipulation Language) ist eine ACL ähnlich KIF, die in dieser Hinsicht erweitert wurde. Das erste Wort einer Nachricht bestimmt den Typ der Kommunikation.
Obwohl KQML im Prinzip nur Feststellungen und Anweisungen unterscheidet, so könnte man diese Typen grundsätzlich in sieben Kategorien einteilen und eine ACL dahingehend erweitern [4] (siehe 1.4):

- *Feststellungen (assertives)*, mit der Aufgabe zu informieren: "Die Tür ist geschlossen."
- *Anweisungen (directives)*, welche um etwas ersuchen: "Schließ die Tür!" - bzw. anfragen *(queries)*: "Können Pelikane fliegen?"
- *Verpflichtungen (commissives)*, die etwas versprechen: "Ich werde die Türe schließen."
- *Erlaubnisse (permissives)*, die Erlaubnis erteilen: "Du kannst die Türe schließen."
- *Verbote (prohibities)*, die eine Aktion untersagen: "Du darfst die Türe nicht schließen."
- *Deklarationen (declaratives)*, welche Ereignisse in sich verursachen: "Ich nenne diese Tür die Golden Gate"
- *Äußerungen (expressives)*, um Emotionen und Einschätzungen auszudrücken: "Ich wünschte, diese Tür wäre geschlossen."

2.3 Architektur von MAS

Sobald man eine ACL und die Möglichkeit, Agenten zu bauen, hat, braucht man nur noch eine funktionierende Organisation zwischen ihnen. Zwei verschieden Ansätze wurden dafür erforscht [3]:

- direkte Kommunikation und

- assistierte Koordination.

Für direkte Kommunikation gibt es die Möglichkeit einer Handlung per Vertrag *(contract net approach)* oder über ein Weitergeben von Spezifikationen *(specification sharing approach)*.
Im "contract net approach" verschicken Agenten, die keine Arbeit haben, Ansuchen um Dienste. Die Empfänger können diese dann auswerten und Angebote zurückschicken. Die ursprünglichen Agenten wiederum können dann entscheiden, welchen Job sie annehmen.
Im "specification sharing approach" schicken arbeitslose Agenten einfach Informationen über ihre Fähigkeiten und Bedürfnisse an andere Agenten, die diese dann auswerten können.
Direkte Kommunikation ist in grossen Systemen, wie zum Beispiel dem Internet, jedoch viel zu kostenintensiv. Und auch die Agenten selbst sind sehr komplex, da sie alle in dieser Form mit anderen Agenten kommunizieren können müssen.

Bei assistierter Koordination kommunizieren Agenten nur mit lokalen Systemprogrammen, sogenannten *facilitators*, und diese dann mit anderen "facilitators", welche wiederum mit "ihren eigenen" Agenten kommunizieren können. Dies zeigt auch schon, daß Agenten in einem solchen System (genannt *federated system*) ihre Autorität an ihre "facilitators" übergeben, welche somit für die Erfüllung ihrer Bedürfnisse verantwortlich sind.

2.4 OSI-Referenz-Modell

W. Bergt definiert in seinem Online-Lexikon [12] den Begriff *OSI-Referenz-Modell* folgendermaßen:

> Ein von der ISO entwickeltes Kommunikationsprotokoll, das allgemeine Festlegungen/Regeln für die Kommunikation in Netzwerken enthält. Das OSI-Referenz-Modell (auch OSI-Schichtenmodell) ist in sieben Schichten (sog. Layer) organisiert, die man in drei sog. Anwendungsschichten und in vier sog. Transportschichten unterteilen kann. Die Anwendungsschichten enthalten Regeln für den Aufbau und den Abbau einer Sitzung, für die Datendarstellung und für anwendungsnahe Grundfunktionen (z. B. Daten- und Dateiübertragung). Die Transportschichten enthalten Regeln für den physikalischen Aufbau einer Sitzung, für die Synchronisation einer Verbindung, für die Wegbestimmung (Routing) und für den logischen Verbindungsaufbau.

Die sieben Schichten werden folgendermaßen beschrieben:

7. Anwendungsschicht Umfaßt sowohl die Netzwerk-Services (Datei-, Druck-, Nachrichten-, Anwendungs- und Datenbank-Service) als auch die dazu benötigten Regeln.

6. Darstellungsschicht Konvertierung von Daten in beiderseitig akzeptiertes Format (z.B. ASCII $\rightarrow$ ANS.1) und Verschlüsselung / Entschlüsselung.

5. Sitzungsschicht (auch Kommunikationssteuerungsschicht) Erleichtert eine Wiederaufnahme einer abgebrochenen Kommunikation an bestimmten Punkten *token* Aufbau und Erhaltung der Verbindung.

4. Transportschicht Verbindet die Netzwerkstrukturen mit den Strukturen der höheren Schichten. Sie unterteilt die Nachrichten der höheren Schichten in Segmente und liefert die Segmente wieder zuverlässig an die Sitzungsschicht. Gebräuchliche Transportprotokolle sind TCP, SPX, NWLink und NetBEUI.

3. Netzwerkschicht Organisation des Austausches von Daten zwischen bestimmten Orten des Netzwerks durch Vermittlung, Netzwerkschicht-Adressierung und Leitweg-Algorithmen. (einen Kommunikationsport festlegen)

2. Datensicherungsschicht in logische Gruppen von Informationen, die Rahmen genannt werden, Fehlerfeststellung, Steuern des Datenflusses im Netzwerk.

1. Bitübertragungsschicht (auch Physikalische Schicht) Koordination der Regeln für die Übertragung der Bits. Die Schicht legt folgendes fest:

- Physische Netzwerkstrukturen,
- Mechanische und elektrische Spezifikationen für die Benutzung des bertragungsmediums
- Bitbertragungscodierung und Timing-Regeln

Nun soll versucht werden, Kommunikationsmodelle bzw. hier die *natürliche Kommunikation* zwischen Menschen auf das OSI-Schichtenmodell abzubilden.

Ablauf auf der Senderseite:

7. Anwendungsschicht Der Kommunikationspartner wird identifiziert.

6. Darstellungsschicht Es wird ein beiderseitig akzeptiertes Format der Kommunikation gesucht. Zum Beispiel wird Englische Sprache festgelegt.

5. Sitzungsschicht (auch Kommunikationssteuerungsschicht) Einbringen von Pausen (token), um die Reaktion des Gesprächspartners abzuwarten

4. Transportschicht Transport der Information durch mündliche Überlieferung

3. Netzwerkschicht entfällt - es kann kein Port bestimmt werden, da der Schall sich in jede Richtung verbreitet ausgehende Nachrichten zu Stimmbändern, eingehende zum Kortex geroutet.

2. Datensicherungsschicht Fehlerfeststellung ist auf Senderseite nicht erforderlich, da der Wortschatz viele Nicht-Code-Wörter enthält

1. Bitübertragungsschicht (auch Physikalische Schicht) Übertragung der Informationen über das Medium Luft mit technischen Daten wie Schallgeschwindigkeit

Ablauf auf der Empfängerseite:

1. Bitübertragungsschicht Aufnahme der akustischen Signale

2. Datensicherungsschicht Überprüfen auf Sinnhaftigkeit (Trennung von Nicht-Code-Wörtern vom Wortschatz)

3. Netzwerkschicht entfällt

4. Transportschicht entfällt

5. Sitzungsschicht Pausen können für Reaktionen auf erkannte Fehler verwendet werden, um die Nachricht zu wiederholen

Anhang A

Protokolle

A.1 9. 3. 1999, 12:30 - 13:00

Ort: Universität Salzburg, Institut für Computerwissenschaften und Systemanalyse, Büro von R. Schwaiger
Teilnehmende Personen: R. Schwaiger, M. Noisternig
Protokollführer: M. Noisternig

Inhalt:

Grundsätzlicher Aufbau des zu erstellenden Dokuments (in LaTeX):

- Titelseite
- Abstract
- Einleitung zu den Fragen “Was ist ein Agent?” und “Was ist Kommunikation in diesem Zusammenhang?”
- Literaturverzeichnis (BIB-File)

Festlegung der Abgabe des Dokuments bis spätestens Montag 13:00 per E-Mail.

A.2 16. 3. 1999, 12:30 - 13:00

Ort: Universität Salzburg, Institut für Computerwissenschaften und Systemanalyse, Büro von R. Schwaiger
Teilnehmende Personen: R. Schwaiger, M. Amersdorfer, M. Noisternig
Protokollführer: M. Amersdorfer, M. Noisternig

Inhalt:

Besprechung des ersten Dokument-Entwurfs.

Zu erledigende Aufgaben:

- Unterschied zwischen Objekt (im Sinne von OO-Programmierung) und Agent
- Anwendungen für Agenten
- Klassifizierung für Agenten
- Muß ein Agent kommunizieren

A.3 23. 3. 1999, 12:30 - 13:00

Ort: Universität Salzburg, Institut für Computerwissenschaften und Systemanalyse, Büro von R. Schwaiger
Teilnehmende Personen: R. Schwaiger, M. Amersdorfer, M. Noisternig
Protokollführer: M. Amersdorfer, M. Noisternig

Inhalt:

Besprechung des bisherigen Dokumententwurfs.

Zu erledigende Aufgaben:

- Korrigieren des Dokuments
- Online-Referenzen für Webster
- Was sind Hardware-Agenten?
- Was sind Multi-Agenten-Systeme?
- Schematische Darstellung eines Agenten, mit internem Status, Zustaende, Schnittstellen, etc.
  Bzw: wie kann man einen Agenten graphisch darstellen?
- Kapitel Kommunikation
  - Welche Ansaetze gibt es?
  - Was ist KQML/KIF, kleiner Überblick und Unterschiede
  - Einsatz von ANN, GA, Fuzzy Logic in Kommunikation
  - Was ist das ISO-OSI Schichtenmodell und wie können Kommunikationsmodelle (natuerliche Kommunikation) darauf abgebildet werden?
    Bzw: Beispiel für natürliche Kommunikation (zwischen Menschen) in OSI-Modell zuordnen versuchen (Netzwerkperspektive)
  - Message-Passing (Bringschuld) - Blackboarding (Holdschuld)

Literaturverzeichnis

[1] Merriam-Webster, *http://www.m-w.com/dictionary.htm.*

[2] T. W. Dictionary, *http://elza.lpi.ru/.*

[3] M. R. Genesereth and S. P. Ketchpel, *Software Agents.* Standford University.

[4] M. P. Singh, *Agent Communication Languages: Rethinking the Principles.* North Carolina State University, 1998.

[5] A. derzeit nicht erhebbar, *What are agents?* http://www.ee.mcgill.ca/~belmarc/UAA.what_are_agents.html.

[6] S. Franklin and A. Graesser, *Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents.* University of Memphis, Insititute for Intelligent Systems, http://www.msci.memphis.edu/~franklin/AgentProg.html.

[7] P. Csurgay, *Intelligent Agents for Telecom Applications.* ITEM/NTNU, http://www.item.ntnu.co/~csurgay/eunice/index.htm.

[8] B. of Rural Sciences, *http://www.brs.gov.au/greenhse/kyoto/hood2.html.* 1998.

[9] Software-Agents-Group, *http://agents.www.media.mit.edu/groups/agents/.*

[10] C. Ranze and J. Mueller, *Ueber den Einsatz von Agenten in Umweltanwendungen.* http://www.informatik.uni-bremen.de/~kcr/paper/UI97/version5.html, 1998.

[11] T. Finin, *http://www.csee.umbc.edu/kse/.* University of Maryland Baltimore County.

[12] W. Bergt, *http://members.aol.com/WBLexikon/lexstart.htm.* 1999.