Agenten in aller Munde: Grundlagen und Hintergründe

Dr. Klaus Dorer, Director Technology Research living systems AG
Christian Dannegger, CTO living systems AG

Abstract

Mit der steigenden Komplexität von Softwaresystemen wird es zunehmend schwieriger, performante, wartungsfähige und funktionierende Software zu erstellen. Ein neues Paradigma soll helfen, diesen Problemen entgegenzuwirken: Agenten. Diese elektronischen Assistenten können selbständig und proaktiv im vorgegebenen Rahmen Entscheidungen treffen und somit ihrem Besitzer Routineaufgaben abnehmen. Wesentliche Eigenschaften von Agenten sind Autonomie, Kollaboration und Mobilität, also die Fähigkeit, die Umgebung zu wechseln, um z.B. im Internet auf einen anderen Server zu migrieren. Dieser Artikel gibt Aufschluss über die Entstehungsgeschichte von Agenten, über deren besonderen Eigenschaften und den aktuellen Stand der Forschung.

Drei weitere Beiträge, die in den Folgemonaten veröffentlicht werden, zeigen die Anwendung der Agententechnologie in Java, ein Anwendungs- und Demonstrationsbeispiel in der Wissenschaft, sowie den Einsatz von Agenten im Produktivumfeld auf.

Historie

Der Begriff des Agenten (engl. agent – im Auftrag Handelnder) wurde Mitte der 80er Jahre in der künstlichen Intelligenz Forschung (KI) geprägt. Damals bildete sich ausgehend vom Massachusetts Institute of Technology (MIT) um Rodney Brooks eine Gegenbewegung zur symbolischen KI, die nicht das abstrakte Schlussfolgern, sondern die Einbettung eines Systems in eine Umgebung, das Wahrnehmen und Handeln in dieser Umgebung in den Mittelpunkt rückte. Mit Hilfe einfacher, reaktiver Algorithmen konnten ihre Roboter erstaunlich schnell relativ komplexe Aufgaben erledigen. Zunächst noch auf Roboter und deren Simulation beschränkt, zeigte sich bald, dass dieses Paradigma autonomer, kollaborativer und situierter, d.h. in eine Umgebung eingebetteter Agenten, für die Modellierung komplexer Systeme hervorragend geeignet ist. Unzählige Publikationen wurden seither veröffentlicht, Workshops im Rahmen von Konferenzen abgehalten und eigene Konferenzen für Agenten eingerichtet (Agents [1], ICMAS [2], PAAM [3] und andere [4]).

Eigenschaften

Auch wenn bis heute die Diskussion anhält, wann ein Programm als Agent bezeichnet werden kann (siehe auch [5, 6]), gelten doch drei Eigenschaften von Agenten als wesentlich: Autonomie, Kollaboration und Mobilität.

Autonomie

Ein wesentlicher Unterschied zu Objekten objektorientierter Software ist, dass Agenten, im vom Benutzer abgesteckten Rahmen, selbständig entscheiden können. Dabei werden grundsätzlich zwei Möglichkeiten der Verhaltensauswahl unterschieden (siehe Abb. 1):

living systems AG - Tel.: +49 (0)771/8987-0 - Fax: +49 (0)771/8987-100 – www.living-systems.com

Reaktives Verhalten ist dann gegeben, wenn der Agent ausschließlich aufgrund der aktuellen Wahrnehmungen auf Änderungen in der Umgebung reagiert. Der Vorteil von reaktivem Verhalten ist, dass es einfach zu implementieren ist und schnell auf Änderungen in der Umgebung reagiert werden kann. Es wird daher bevorzugt in dynamischen, d.h. sich schnell ändernden Umgebungen eingesetzt.

Demgegenüber gilt Verhalten als proaktiv, wenn es vorausschauend und zielgerichtet ist. Proaktives Verhalten verlangt also mindestens die explizite Repräsentation der Ziele des Agenten. Idealerweise ist ein Agent in der Lage, sowohl reaktiv als auch proaktiv zu agieren. Diese nicht triviale Aufgabe, reaktives und proaktives Verhalten zu integrieren, ist seit der Entstehung von Agententechnologie Gegenstand der Forschung.

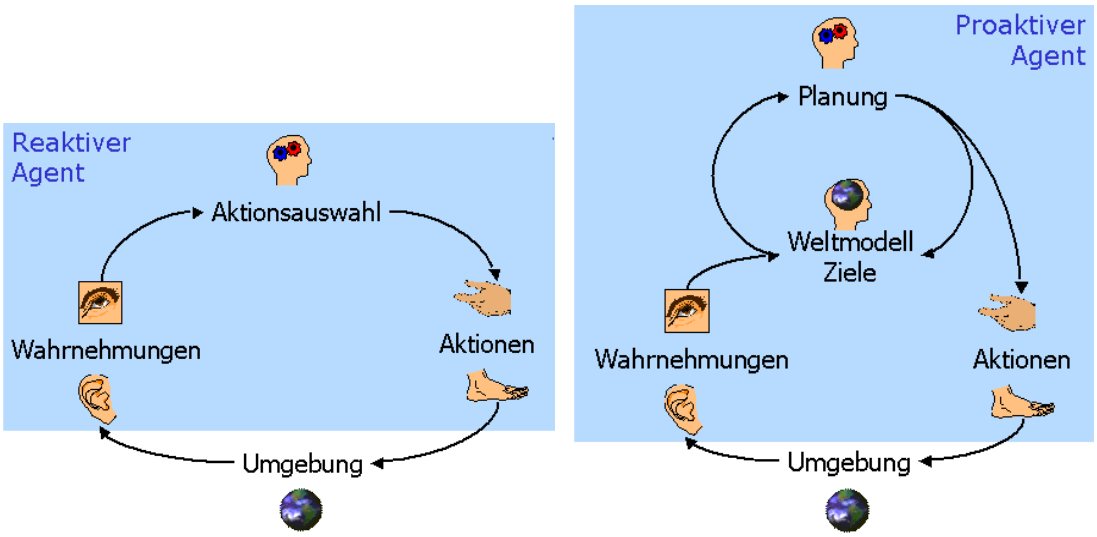

Abbildung 1: Reaktive (links) und proaktive (rechts) Verhaltenskontrolle.

Kollaboration

Agentensysteme sind dadurch charakterisiert, dass viele Agenten miteinander interagieren. Das kann sowohl indirekt durch die Änderung der gemeinsamen Umgebung geschehen, wenn beispielsweise Fußballagenten den Ball hin und her spielen, als auch direkt mittels Kommunikation. Um eine reibungslose Kommunikation unterschiedlicher Agenten zu gewährleisten, wurden verschiedene Standards für Agentenkommunikation eingeführt. Zu den verbreitetsten zählen KQML, KIF sowie FIPA-ACL (siehe [7]). Im Unterschied zur herkömmlichen Programmierung kann aber der Agent entscheiden, wann er mit wem und ob er überhaupt kommuniziert [8]. Auch bei der Kollaboration von Agenten gilt, dass das Ganze mehr ist als die Summe seiner Teile. Sprich, erst durch die Zusammenarbeit von Agenten können komplexere Aufgaben wie etwa eine Auktion bewältigt werden. Genau wie in realen Organisationen schafft diese Eigenschaft gleichzeitig die Herausforderung, die Zusammenarbeit so zu organisieren, dass eine optimale Gesamtperformance entsteht [6].

Mobilität

Agenten sind in der Lage, ihre Umgebung (Server) zu verlassen und zu einem neuen Server zu migrieren. Im Unterschied zum Austausch von Nachrichten (data-shipping) wird hier sowohl der Zustand als auch die Funktionalität des Agenten verschickt (code-shipping). Diese Mobilität wird beispielsweise in der

living systems AG - Tel.: +49 (0)771/8987-0 - Fax: +49 (0)771/8987-100 – www.living-systems.com

Informationssuche genutzt, bei der Agenten von Server zu Server ziehen, um interessante Daten zu filtern und später mit der gefilterten Information zum Auftraggeber der Suche zurückzukehren. Dadurch kann vermieden werden, dass alle Daten über das Netzwerk geschickt werden müssen, von denen anschließend nur ein winziger Bruchteil verwendet wird. Andere Anwendungen liegen in der freien Wahl eines Marktplatzes bzw. im Wechsel zwischen Marktplätzen im Internet oder im Verlassen eines Servers, der beispielsweise gewartet werden muss.

Agentenorientiertes Software Engineering

Inzwischen wurde das Paradigma ‚Agenten' auch um definierte Methoden und Prozesse erweitert, die systematisch den Prozess der Planung und Erstellung von Agentensystemen unterstützen. Neben Erweiterungen objektorientierter Methoden wie z.B. KGR (benannt nach den Autoren Kinny, Georgeff und Rao) oder Agent-UML (siehe auch [8]), wurden auch rein agentenspezifische Methoden der Modellierung vorgeschlagen. Ein Beispiel ist Gaia [9], anhand dessen Beschreibung auch die Spezifika von agentenorientiertem Software Engineering verdeutlicht werden sollen. Gaia stellt für Analyse und Design verschiedene Modelle zur Verfügung (siehe Abb. 2), um die Anforderungen an ein System agentenorientiert zu strukturieren und für die Umsetzung mit objektorientierten Methoden vorzubereiten.

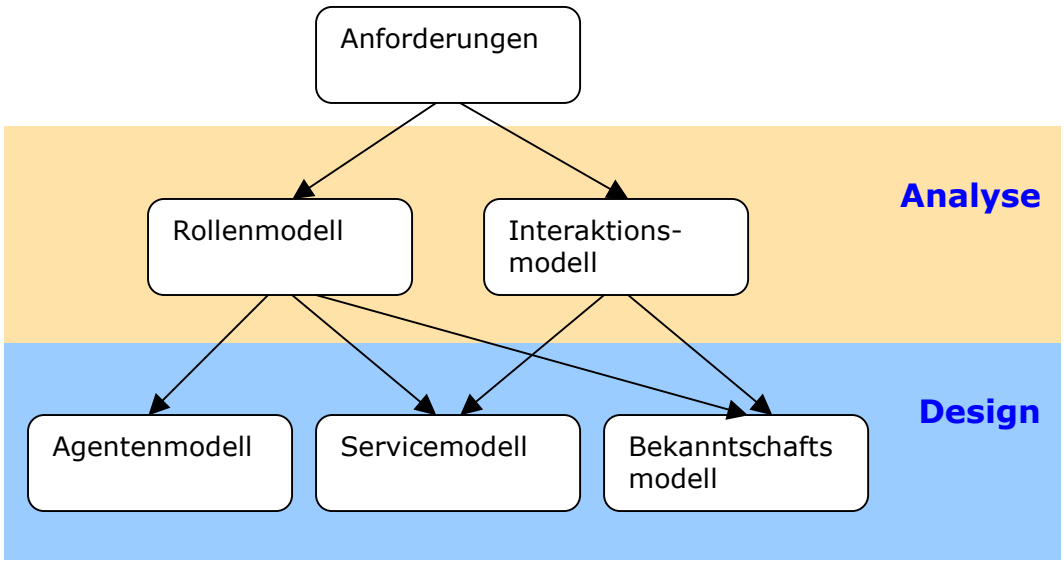

Abbildung 2: Beziehungen zwischen den einzelnen Gaia-Modellen (aus [9])

Analyse

Üblicherweise wird ein System als Organisation aufgefasst, das wie in ‚realen' Organisationen durch die Definition von Rollen strukturiert wird. Im Rollenmodell wird dementsprechend dokumentiert, welche Rollen nötig sind (z.B.: Firmenleiter, Verkaufsleiter, ...), welche Verantwortlichkeiten einer Rolle zugewiesen werden (z.B. Akquisition) und welche Rechte eine Rolle besitzt (z.B. Prokura oder das Recht Gehaltserhöhungen auszusprechen). Das Interaktionsmodell beschreibt die Abhängigkeiten und Beziehungen zwischen den einzelnen Rollen. Es legt also fest, welche Rollen miteinander kommunizieren sowie die Protokoll-Definitionen, die

living systems AG - Tel.: +49 (0)771/8987-0 - Fax: +49 (0)771/8987-100 – www.living-systems.com

unter anderem Zweck, Initiator, Inputs, Outputs der Kommunikation nicht aber den Ablauf oder die Abfolge von Nachrichten definieren.

Design

Das Ziel des Designs in Gaia ist eine Spezifikation, die detailliert genug ist, um darauf aufbauend mit objektorientierten Techniken einzelne Agenten zu entwerfen und zu implementieren. Es wird also nur spezifiziert, wie mit Hilfe von Agenten die Systemziele erreicht werden können und welche Aufgaben die einzelnen Agenten dabei übernehmen, nicht aber, wie ein einzelner Agent diese Aufgaben realisiert. Für das Design werden in Gaia drei Modelle vorgeschlagen: Agentenmodell, Servicemodell und Bekanntschaftsmodell.

Das Agentenmodell legt fest, welche Agenten im System vorhanden sind und welche Rollen diese übernehmen. Mehrere Rollen können (aus Effizienzgründen) in einem Agenten zusammengefasst werden genauso wie eine Rolle von mehreren Agenten eingenommen werden kann, um z.B. bei erwarteter hoher Arbeitslast entsprechend skalierbar zu sein.

Das Servicemodell definiert die Services, die von Agenten zur Verfügung gestellt werden. Ein Service ist definiert durch die Inputs und Outputs, die sich aus den Protokolldefinitionen der Analysephasen ergeben, sowie durch die Vor- und Nachbedingungen, die durch die Verantwortlichkeiten und Rechte von Rollen definiert sind. Beispiel für einen Service wäre die Durchführung einer Gehaltserhöhung durch einen Abteilungsleiter.

Das Bekanntschaftsmodell schließlich ist ein Graph, der darstellt, welche Agenten untereinander kommunizieren. Es soll insbesondere helfen, Engpässe und Flaschenhälse in der Kommunikation aufzudecken [9].

Fazit

Agententechnologie ist längst den Kinderschuhen entwachsen. Die Analysten des Research-Hauses IDC (International Data Corporation) beispielsweise sehen den Markt für Agententechnologie stark wachsen. Von 112 Mio. US $ im Jahr 2000 prophezeien sie einen Anstieg auf annähernd 900 Mio. US $ in 2004. Mit ihren speziellen Eigenschaften können Agenten sehr flexibel gerade auch in dynamischen und kollaborativen Umgebungen eingesetzt werden. Agentenorientiertes Software Engineering stellt die Methoden zur Verfügung, mit Hilfe derer Agentensysteme entworfen werden können.

Konferenzen

[1] Autonomous Agents, jährlich seit 1997, http://autonomousAgents.org/
[2] ICMAS, International Conference on Multi Agent Systems, erstmals 1995, seit 1996 alle zwei Jahre, ICMAS-2000 unter http://www.computer.org/proceedings/icmas/0625/0625toc.htm
[3] PAAM, Practical Application of Intelligent Agents and Multi-Agents, jährlich seit 1996, http://www.practical-applications.co.uk/PAAM/index.html
[4] Ein umfangreiches Verzeichnis zum Thema Agenten bietet Leonardo Garrido unter http://www-cia.mty.itesm.mx/~lgarrido/Repositories/IA/index.html

living systems AG - Tel.: +49 (0)771/8987-0 - Fax: +49 (0)771/8987-100 – www.living-systems.com

Referenzen

[5] Franklin, S., and Graesser, A., (1997). Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agent. In: *Intelligent Agents III*, Seiten 21-35, Springer-Verlag:Heidelberg.

[6] Jennings, N. R., and Wooldridge, M. (2001). Agent-Oriented Software Engineering. In: J. Bradshaw (ed.) *Handbook of Agent Technology*, AAAI/MIT Press. (to appear).

[7] Kone, M.T., Shimazu, A., und Nakajima, T. (2000). The State of the Art in Agent Communication Languages. *Knowledge and Information Systems 2*, Seiten 259-284.

[8] Weiß, G. (2001) Agentenorientiertes Software Engineering. *Informatik Spektrum*, Band 24, Heft 2, Seiten 98-101, Springer-Verlag:Heidelberg.

[9] Wooldridge, M., Jennings, N. R., and Kinny, D. (2000) The Gaia Methodology for Agent-Oriented Analysis and Design. In: *Journal of Autonomous Agents and Multi-Agent Systems* 3 (3) Seiten 285-312.

living systems AG - Tel.: +49 (0)771/8987-0 - Fax: +49 (0)771/8987-100 – www.living-systems.com