Software agent

From Wikipedia, the free encyclopedia

In computer science, a **software agent** is a computer program that acts for a user or other program in a relationship of agency, which derives from the Latin *agere* (to do): an agreement to act on one's behalf. Such "action on behalf of" implies the authority to decide which, if any, action is appropriate.[1][2] Agents are colloquially known as *bots*, from *robot*. They may be embodied, as when execution is paired with a robot body, or as software such as a chatbot executing on a phone (e.g. Siri) or other computing device. Software Agents may be autonomous or work together with other agents or people. Software agents interacting with people (e.g. chatbots, human-robot interaction environments) may possess human-like qualities such as natural language understanding and speech, personality or embody humanoid form (see Asimo).

Related and derived concepts include *intelligent agents* (in particular exhibiting some aspect of artificial intelligence, such as learning and reasoning), *autonomous agents* (capable of modifying the way in which they achieve their objectives), *distributed* agents (being executed on physically distinct computers), *multi-agent systems* (distributed agents that work together to achieve an objective that could not be accomplished by a single agent acting alone), and *mobile agents* (agents that can relocate their execution onto different processors).

Contents

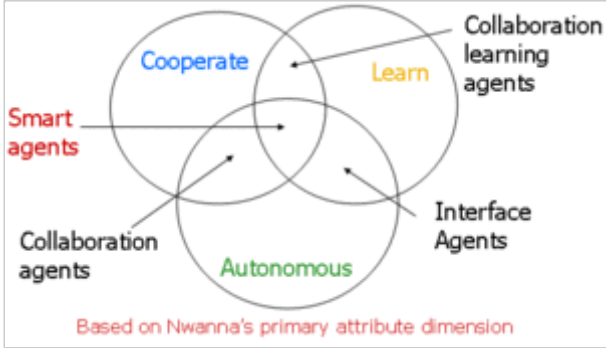
Concepts

The basic attributes of an autonomous software agent are that agents

- are not strictly invoked for a task, but activate themselves,
- may reside in wait status on a host, perceiving context,

- may get to run status on a host upon starting conditions,
- do not require interaction of user,
- may invoke other tasks including communication.

The term "agent" describes a software abstraction, an idea, or a concept, similar to OOP terms such as methods, functions, and objects. The concept of an agent provides a convenient and powerful way to describe a complex software entity that is capable of acting with a certain degree of autonomy in order to accomplish tasks on behalf of its host. But unlike objects, which are defined in terms of *methods* and *attributes*, an agent is defined in terms of its behavior[3].



Nwana's Category of Software Agent

Various authors have proposed different definitions of agents, these commonly include concepts such as

- *persistence* (code is not executed on demand but runs continuously and decides for itself when it should perform some activity)
- *autonomy* (agents have capabilities of task selection, prioritization, goal-directed behaviour, decision-making without human intervention)
- *social ability* (agents are able to engage other components through some sort of communication and coordination, they may collaborate on a task)
- *reactivity* (agents perceive the context in which they operate and react to it appropriately).

Distinguishing agents from programs

All agents are programs, but not all programs are agents. Contrasting the term with related concepts may help clarify its meaning. Franklin & Graesser (1997)[4] discuss four key notions that distinguish agents from arbitrary programs: reaction to the environment, autonomy, goal-orientation and persistence.

Intuitive distinguishing agents from objects

- Agents are more autonomous than objects.
- Agents have flexible behaviour: reactive, proactive, social.
- Agents have at least one thread of control but may have more.[5]

Distinguishing agents from expert systems

- Expert systems are not coupled to their environment.
- Expert systems are not designed for reactive, proactive behavior.
- Expert systems do not consider social ability.[5]

Distinguishing intelligent software agents from intelligent agents in artificial intelligence

- Intelligent agents (also known as rational agents) are not just computer programs: they may also be machines, human beings, communities of human beings (such as firms) or anything that is capable of goal directed behavior.

  (Russell & Norvig 2003)

Impact of software agents

Software agents may offer various benefits to their end users by automating complex or repetitive tasks.[6] However, there are organizational and cultural impacts of this technology that need to be considered prior to implementing software agents.

Organizational impact

Organizational impacts include the transformation of the entire electronic commerce sector, operational encumbrance, and security overload. Software agents are able to quickly search the Internet, identify the best offers available online, and present this information to the end users in aggregate form. Therefore, users may not need to manually browse various websites of individual merchants; they are able to locate the best deal in a matter of seconds. At the same time, this increases price-based competition and transforms the entire electronic commerce sector into a uniform perfect competition market. The implementation of agents also requires additional resources from the companies, places an extra burden on their networks, and requires new security process.

Software agents are also increasingly used to enter data and query databases at a business, as the traditional paper form and its processing at a business become automated into one or more distinct databases maintained at a business. In the twenty-first century, software agents are expected to take on routine manual database related tasks that a human would otherwise waste time performing manually at a business as businesses increasingly glue together many different software graphical tools and their underlying databases.

Work contentment and job satisfaction impact

People like to perform easy tasks providing the sensation of success unless the repetition of the simple tasking is affecting the overall output. In general implementing software agents to perform administrative requirements provides a substantial increase in work contentment, as administering their own work does never please the worker. The effort freed up serves for higher degree of engagement in the substantial tasks of individual work. Hence, software agents may provide the basics to implement self-controlled work, relieved from hierarchical controls and interference.[7] Such conditions may be secured by application of software agents for required formal support.

Cultural impact

The cultural effects of the implementation of software agents include trust affliction, skills erosion, privacy attrition and social detachment. Some users may not feel entirely comfortable fully delegating important tasks to software applications. Those who start relying solely on intelligent agents may lose important skills, for example, relating to information literacy. In order to act on a user's behalf, a software agent needs to have a complete understanding of a user's profile, including his/her personal preferences. This, in turn, may lead to unpredictable privacy issues. When users start relying on their software agents more, especially for communication activities, they may lose contact with other human users and look at the word with the eyes of their agents. These consequences are what agent researchers and users must consider when dealing with intelligent agent technologies.[8]

History

The concept of an agent can be traced back to Hewitt's Actor Model (Hewitt, 1977) - "A self-contained, interactive and concurrently-executing object, possessing internal state and communication capability."

To be more academic, software agent systems are a direct evolution from Multi-Agent Systems (MAS). MAS evolved from Distributed Artificial Intelligence (DAI), Distributed Problem Solving (DPS) and Parallel AI (PAI), thus inheriting all characteristics (good and bad) from DAI and AI.

John Sculley's 1987 "Knowledge Navigator" video portrayed an image of a relationship between end-users and agents. Being an ideal first, this field experienced a series of unsuccessful top-down implementations, instead of a piece-by-piece, bottom-up approach. The range of agent types is now (from 1990) broad: WWW, search engines, etc.

Examples of intelligent software agents

Haag (2006) suggests that there are only four essential types of intelligent software agents:[9]

1. Buyer agents or shopping bots
2. User or personal agents
3. Monitoring-and-surveillance agents
4. Data-mining agents

Buyer agents (shopping bots)

Buyer agents travel around a network (e.g. the internet) retrieving information about goods and services. These agents, also known as 'shopping bots', work very efficiently for commodity products such as CDs, books, electronic components, and other one-size-fits-all products.

User agents (personal agents)

User agents, or personal agents, are intelligent agents that take action on your behalf. In this category belong those intelligent agents that already perform, or will shortly perform, the following tasks:

- Check your e-mail, sort it according to the user's order of preference, and alert you when important emails arrive.
- Play computer games as your opponent or patrol game areas for you.
- Assemble customized news reports for you. There are several versions of these, including CNN.
- Find information for you on the subject of your choice.
- Fill out forms on the Web automatically for you, storing your information for future reference
- Scan Web pages looking for and highlighting text that constitutes the "important" part of the information there
- Discuss topics with you ranging from your deepest fears to sports
- Facilitate with online job search duties by scanning known job boards and sending the resume to opportunities who meet the desired criteria
- Profile synchronization across heterogeneous social networks

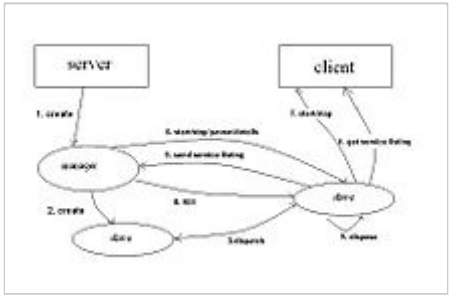
Monitoring-and-surveillance (predictive) agents

Monitoring and Surveillance Agents are used to observe and report on equipment, usually computer systems. The agents may keep track of company inventory levels, observe competitors' prices and relay them back to the company, watch stock manipulation by insider trading and rumors, etc.

For example, NASA's Jet Propulsion Laboratory has an agent that monitors inventory, planning, and scheduling equipment ordering to keep costs down, as well as food storage facilities. These agents usually monitor complex computer networks that can keep track of the configuration of each computer connected to the network.

A special case of Monitoring-and-Surveillance agents are organizations of agents used to emulate the Human Decision Making process during tactical operations. The agents monitor the status of assets (ammunition, weapons available, platforms for transport, etc.) and receive Goals (Missions) from higher level agents. The Agents then

pursue the Goals with the Assets at hand, minimizing expenditure of the Assets while maximizing Goal Attainment. (See Popplewell, "Agents and Applicability")



service monitoring

Data-mining agents

This agent uses information technology to find trends and patterns in an abundance of information from many different sources. The user can sort through this information in order to find whatever information they are seeking.

A data mining agent operates in a data warehouse discovering information. A 'data warehouse' brings together information from lots of different sources. "Data mining" is the process of looking through the data warehouse to find information that you can use to take action, such as ways to increase sales or keep customers who are considering defecting.

'Classification' is one of the most common types of data mining, which finds patterns in information and categorizes them into different classes. Data mining agents can also detect major shifts in trends or a key indicator and can detect the presence of new information and alert you to it. For example, the agent may detect a decline in the construction industry for an economy; based on this relayed information construction companies will be able to make intelligent decisions regarding the hiring/firing of employees or the purchase/lease of equipment in order to best suit their firm.

Networking and communicating agents

Some other examples of current intelligent agents include some spam filters, game bots, and server monitoring tools. Search engine indexing bots also qualify as intelligent agents.

- User agent - for browsing the World Wide Web
- Mail transfer agent - For serving E-mail, such as *Microsoft Outlook*. Why? It communicates with the POP3 mail server, without users having to understand POP3 command protocols. It even has rule sets that filter mail for the user, thus sparing them the trouble of having to do it themselves.
- SNMP agent
- In Unix-style networking servers, *httpd* is an HTTP daemon that implements the Hypertext Transfer Protocol at the root of the World Wide Web
- Management agents used to manage telecom devices
- Crowd simulation for safety planning or 3D computer graphics,
- Wireless *beaconing agent* is a simple process hosted single tasking entity for implementing wireless lock or electronic leash in conjunction with more complex software agents hosted e.g. on wireless receivers.
- Use of autonomous agents (deliberately equipped with noise) to optimize coordination in groups online.[10]

Design issues

Issues to consider in the development of agent-based systems include

- how tasks are scheduled and how synchronization of tasks is achieved
- how tasks are prioritized by agents
- how agents can collaborate, or recruit resources,
- how agents can be re-instantiated in different environments, and how their internal state can be stored,
- how the environment will be probed and how a change of environment leads to behavioral changes of the agents
- how messaging and communication can be achieved,

- what hierarchies of agents are useful (e.g. task execution agents, scheduling agents, resource providers ...).

For software agents to work together efficiently they must share semantics of their data elements. This can be done by having computer systems publish their metadata.

The definition of *agent processing* can be approached from two interrelated directions:

- internal state processing and ontologies for representing knowledge
- interaction protocols – standards for specifying communication of tasks

Agent systems are used to model real-world systems with concurrency or parallel processing.

- Agent Machinery – Engines of various kinds, which support the varying degrees of intelligence
- Agent Content – Data employed by the machinery in Reasoning and Learning
- Agent Access – Methods to enable the machinery to perceive content and perform actions as outcomes of Reasoning
- Agent Security – Concerns related to distributed computing, augmented by a few special concerns related to agents

The agent uses its access methods to go out into local and remote databases to forage for content. These access methods may include setting up news stream delivery to the agent, or retrieval from bulletin boards, or using a spider to walk the Web. The content that is retrieved in this way is probably already partially filtered – by the selection of the newsfeed or the databases that are searched. The agent next may use its detailed searching or language-processing machinery to extract keywords or signatures from the body of the content that has been received or retrieved. This abstracted content (or event) is then passed to the agent's Reasoning or inferencing machinery in order to decide what to do with the new content. This process combines the event content with the rule-based or knowledge content provided by the user. If this process finds a good hit or match in the new content, the agent may use another piece of its machinery to do a more detailed search on the content. Finally, the agent may decide to take an action based on the new content; for example, to notify the user that an important event has occurred. This action is verified by a security function and then given the authority of the user. The agent makes use of a user-access method to deliver that message to the user. If the user confirms that the event is important by acting quickly on the notification, the agent may also employ its learning machinery to increase its weighting for this kind of event.

Notions and frameworks for agents

- DAML (DARPA Agent Markup Language)
- Jason (multi-agent systems development platform)
- 3APL (Artificial Autonomous Agents Programming Language)
- GOAL agent programming language
- Web Ontology Language (OWL)
- *daemons* in Unix-like systems.
- Java Agent Template (JAT)
- Java Agent Development Framework (JADE)
- SARL agent programming language (arguably an Actor and not Agent oriented paradigm)

See also

- Agent architecture

References

1. Nwana, H. S. (1996). "Software Agents: An Overview". *Knowledge Engineering Review*. Cambridge University Press. **21** (3): 205–244. doi:10.1017/s026988890000789x (https://doi.org/10.1017%2Fs026988890000789x).
2. Schermer,, B. W. (2007). "Software agents, surveillance, and the right to privacy: A legislative framework for agent-enabled surveillance" (https://openaccess.leidenuniv.nl/handle/1887/11951) (paperback). **21** (3). Leiden University Press: 140, 205–244. ISBN 978-0-596-00712-6. hdl:1887/11951 (https://hdl.handle.net/1887%2F11951). Retrieved 2012-10-30.
3. Wooldridge, M.; Jennings, N. R. (1995). "Intelligent agents: theory and practice". **10** (2). Knowledge Engineering Review: 115–152.
4. Franklin, S.; Graesser, A. (1996). "Is it an Agent, or just a Program?: A Taxonomy for Autonomous Agents" (http://www.msci.memphis.edu/~franklin/AgentProg.html). University of Memphis, Institute for Intelligent Systems. Archived from the original (http://www.msci.memphis.edu/) on 1996.
5. Wooldridge, Michael J. (2002). *An Introduction to Multiagent Systems* (http://www.amazon.com/gp/product/047149691X/#reader_047149691X). New York: John Wiley & Sons. p. 27. ISBN 0-471-49691-X.
6. Serenko, A.; Detlor, B. (2004). "Intelligent agents as innovations" (http://www.aserenko.com/papers/Serenko_Detlor_AI_and_Society.pdf) (PDF). **18** (4): 364–381.
7. Adonisi, M. (2003). "The relationship between Corporate Entrepreneurship, Market Orientation, Organisational Flexibility and Job satisfaction" (http://upetd.up.ac.za/thesis/available/etd-11252004-150603/unrestricted/00thesis.pdf) (PDF) (Diss.). Fac.of Econ.and Mgmt.Sci., Univ.of Pretoria.
8. Serenko, A.; Ruhi, U.; Cocosila, M. (2007). "Unplanned effects of intelligent agents on Internet use: Social Informatics approach" (http://www.aserenko.com/papers/AI_Society_Serenko_Social_Impacts_of_Agents.pdf) (PDF). **21** (1-2). Artificial Intelligence & Society: 141–166.
9. Haag, Stephen (2006). "Management Information Systems for the Information Age": 224–228.
10. Shirado, Hirokazu; Christakis, Nicholas A. "Locally noisy autonomous agents improve global human coordination in network experiments" (http://www.nature.com/doifinder/10.1038/nature22332). *Nature*. **545** (7654): 370–374. doi:10.1038/nature22332 (https://doi.org/10.1038%2Fnature22332).

External links

- Software Agents: An Overview (http://agents.umbc.edu/introduction/ao/), Hyacinth S. Nwana. *Knowledge Engineering Review*, 11(3):1–40, September 1996. Cambridge University Press.
- FIPA (http://www.fipa.org/) The Foundation for Intelligent Physical Agents
- JADE (http://jade.tilab.com/) Java Agent Developing Framework, an Open Source framework developed by Telecom Italia Labs
- European Software-Agent Research Center (http://www.software-agent.eu/)
- SemanticAgent (http://code.google.com/p/semanticagent/) An Open Source framework to develop SWRL based Agents on top of JADE
- Mobile-C (http://www.mobilec.org/) A Multi-Agent Platform for Mobile C/C++ Agents.
- HLL (http://isr.nu/hll/project/JavaNetReflect/) High Level Logic (HLL) Open Source Project.
- Open source project KATO (http://kato.sourceforge.net/kato.html) for PHP and Java developers to write software agents