

Intelligent Agents and Distributed Knowledge Base Management Systems

Author: Arman Kanooni

Date: December 1, 2002

Table of Contents

Abstract	3
Keywords	3
Audience	3
Introduction	4
Why do we need agents?.....	4
What Is an Intelligent Agent?	4
A Typology of Agents.....	4
Taxonomy of Agents.....	7
<i>Collaborative Agents</i>	7
<i>Interface Agents</i>	8
<i>Mobile Agents</i>	8
<i>Information/Internet Agents</i>	9
<i>Reactive Software Agents</i>	10
<i>Hybrid Agents</i>	10
<i>Heterogeneous Agents</i>	11
Future of Agents.....	11
Distributed Knowledge Based Management System.....	12
<i>DKBMS Creation</i>	12
<i>Intelligent Agents and DKMS</i>	12
<i>Object/Component Management and Agents</i>	13
<i>Use Case/Workflow Management and Agents</i>	14
Tools.....	15
<i>Search Agents</i>	15
<i>Personal Assistance</i>	15

Abstract

The increasingly dependent upon electronic sources of information and data overload (e-mail, web pages, fax, etc.) and inadequacy of current tools, motivates the research and development in other software solution (agents) that can act in our place. This paper reviews the intelligent agent-based systems taxonomy. The high level architecture of each software agent is described. We will have a high level review of the Distributed Knowledge Based Management System and an analysis of how intelligent agents can participate in DKBMS architecture.

Keywords

Intelligent Agents, Human Agents, Hardware Agents (e. g. robot), Interface Agent, Information Agents, Cooperation Agents, Transaction Agents, Software Agents, Intelligent Agents, **Information Agent, Cooperation Agent, Transaction Agent, DKBMS, AKM, WebSeeker, TextALoud, Spider**

Audience

This document is targeted to people interested in study and development of Distributed Artificial Intelligence and its practical application such as Intelligent Agents.

Introduction

“The *American Heritage Dictionary* defines an agent as “one that acts or has the power or authority to act... or represent another” or the “means by which something is done or caused; instrument.” The term derives from the present participle of the Latin verb *agere*: to drive, lead, act, or do.”¹

“The concept of an agent can be traced back to the early days of research into Distributed Artificial Intelligence in the 1970s - indeed, to Carl Hewitt concurrent Actor model (Hewitt, 1977). In this model, Hewitt proposed the concept of a self-contained, interactive and concurrently executing object, which he termed actor. This object had some encapsulated internal state and could respond to messages from other similar objects: an actor is a computational agent, which has a mail address and a behavior. Actors communicate by message-passing and carry out their actions concurrently (Hewitt, 1977, p. 131).”²

Why do we need agents?

We live in an increasingly networked environment. We are overloaded with data in form of e- mail, web pages, fax, etc. There is a greater exchange of digital information between individuals and businesses. We are increasingly dependent upon electronic sources of information to make personal or business decision.

Current tools are inadequate. World Wide Web is too polluted for casual browsing. The intelligent search tools are required; even search engines beginning to fail us!

We need software solution (agents) that can act in our place: It can interact with Internet data sources. It can process e- mail, voice, fax and other electronic message sources. It can communicate with other agents. It can accurately represent our needs and preferences in the networked information environment.

What Is an Intelligent Agent?

An Intelligent Agent is a "component of software and/or hardware which is capable of acting on behalf of its client and behaves to at least some degree: autonomously (without continuous direction), socially (interacts with other agents), proactively (influences its environment), and reactively (is influenced by its environment)."²

A Typology of Agents

"Firstly, agents may be classified by their mobility, i.e. by their ability to move around some network. This yields the classes of *static* or *mobile* agents.

Secondly, they may be classed as either *deliberative* or *reactive*. Deliberative agents derive from the deliberative thinking paradigm: the agents possess an internal symbolic, reasoning model and they engage in planning and negotiation in order to achieve coordination with other agents.

Thirdly, agents may be classified along several ideal and primary attributes which agents *should* exhibit.

We have identified a minimal list of three: autonomy, learning and cooperation. *Autonomy* refers to the principle that agents can operate on their own without the need for human guidance, even though this would sometimes be invaluable.

Cooperation with other agents is paramount: it is the *raison d'être* for having multiple agents in the first place in contrast to having just one.

Lastly, for agent systems to be truly smart, they would have to *learn* as they react and/or interact with their external environment.

We use these three minimal characteristics in Figure 1 to derive four types of agents to include in our typology: *collaborative agents*, *collaborative learning agents*, *interface agents* and truly *smart agents*." ²

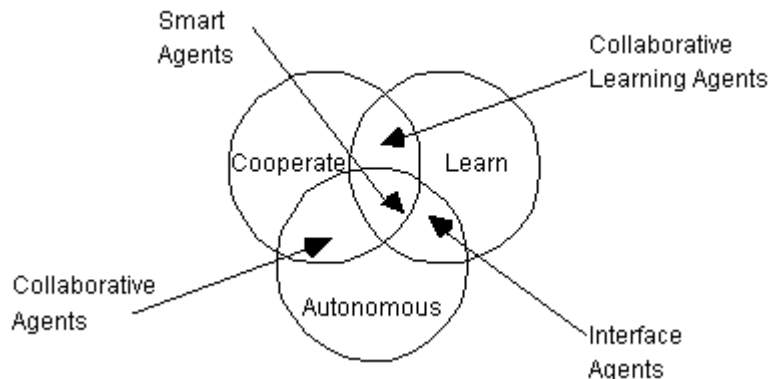


Figure 1 - A Part View of an Agent Typology

Fourthly, agents may sometimes be classified by their roles, e.g. WWW information agents. This category of agents usually exploits Internet search engines such as WebCrawlers, Lycos and Spiders. We refer to these classes of agents as *information* or *Internet agents*.

Fifthly, we have also included the category of *hybrid* agents, which combine of two or more agent philosophies in a single agent.

"We identify seven types of agents:

- Collaborative agents
- Interface agents
- Mobile agents

- Information/Internet agents
- Reactive agents
- Hybrid agents
- Heterogeneous Agents" ²

Taxonomy of Agents

Figure 2 shows an agent typology.²



Figure 2 - A Classification of Software Agents

Collaborative Agents

"Collaborative agents emphasize autonomy and cooperation (with other agents) in order to perform tasks for their owners. In order to have a coordinated set up of collaborative agents; they may have to *negotiate* in order to reach mutually acceptable agreements on some matters."²

The goal for having collaborative agent systems is "creating a system that interconnects separately developed collaborative agents, thus enabling the ensemble to function beyond the capabilities of any of its members".³

"The Pleiades project at CMU directed by Tom Mitchell and Katia Sycara applies collaborative agents in the domain of Organizational Decision Making over the "InfoSphere" (which refers essentially to a collection of internet-based heterogeneous resources)".²

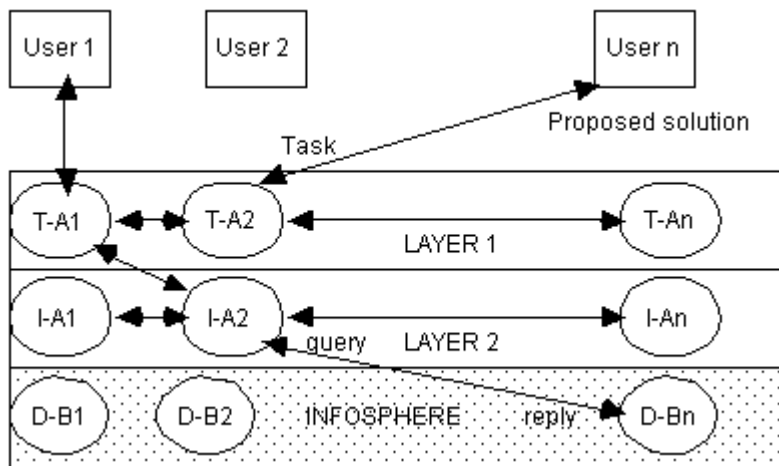


Figure 3 - The Pleiades Distributed System Architecture⁴

"The first layer contains task-specific collaborative agents. The second layer contains *information-specific* collaborative agents. Task-specific agents, depicted as task-assistants (T-A) in the figure, perform a particular task for its user, e.g.

arranging appointments and meetings with other task-specific agents. These agents coordinate and schedule plans based on the context. They collaborate with one another (at level 1) in order to resolve conflicts or integrate information. In order to garner the information required at this level, they request information from information-specific agents, depicted as information assistants (I-A) in Figure 3. Information-specific agents, in turn, may collaborate with one another (i.e. within layer 2) in order to provide the information requested back to the layer 1 requesting agent. The source of the information is the many databases (D-B) in the infosphere. Ultimately, the task agent proposes a solution (sometimes an intermediate one) to its user. Agents communicate using 'KQML' ⁵ and e-mail, and they negotiate in order to reach agreements in cases of conflicts". ²

Interface Agents

Pattie Maes, a key proponent of this class of agents, points out that "the key metaphor underlying interface agents is that of a *personal assistant* who is *collaborating with the user* in the same work environment." ⁶

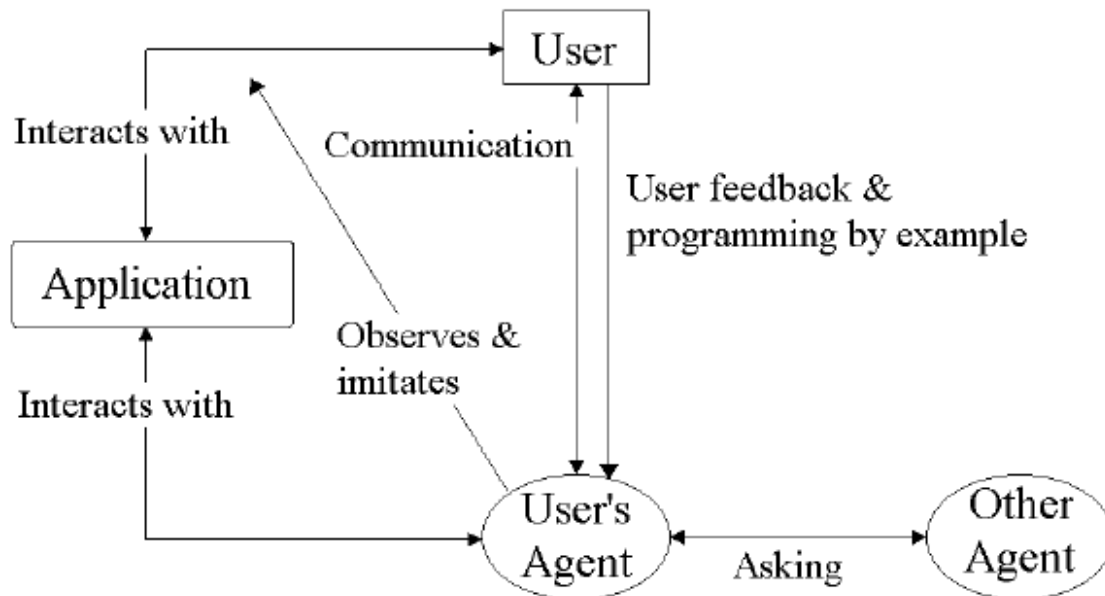


Figure 4 - How Interface Agents Work ⁶

Mobile Agents

"Mobile agents are computational software processes capable of roaming wide area networks (WANs) such as the WWW, interacting with foreign hosts, gathering information on behalf of its owner and coming 'back home' having performed the duties set by its user.

Telescript is an interpreted object-oriented and remote programming language, which allows for the development of distributed applications. The interpreter and runtime development environment for the Telescript language is called the *Telescript engine* and a given host can support simultaneously multiple Telescript engines. Figure 5 summarizes a part view of the Telescript architecture. It shows just one of these Telescript engines integrated onto an operating system via a programming interface called the Telescript application programmer interface (API)."⁷

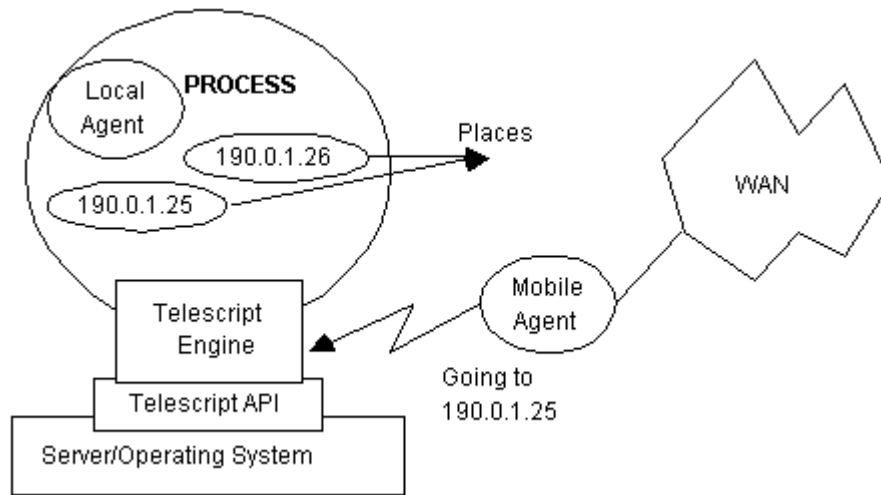


Figure 5 - A Part View of Telescript Architecture⁷

Information/Internet Agents

"Figure 6 shows how an information agent uses a host of internet management tools such as Spiders and search engines in order to gather the information. The information agent may be associated with some particular indexer(s), e.g. a Spider. Other search/indexing engines or spiders such as Lycos or Webcrawler can be used similarly to build up the index."⁸

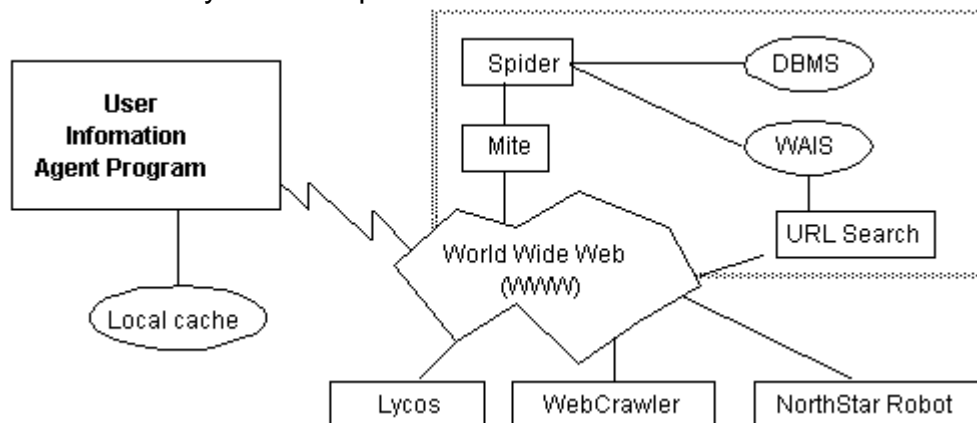


Figure 6 - A view of how Information Agents Work⁸

Reactive Software Agents

"Reactive agents represent a special category of agents, which do *not* possess internal, symbolic models of their environments; instead they act/respond in a stimulus-response manner to the present state of the environment in which they are embedded." ⁹

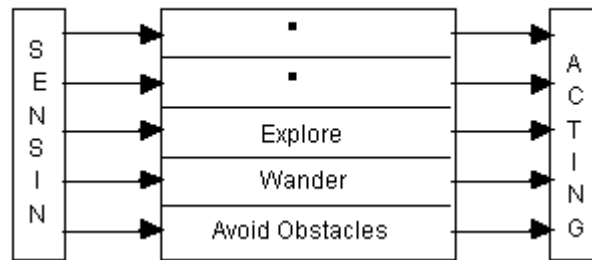


Figure 7 - Brook's Subsumption Architecture ⁹

Hybrid Agents

A prototypical example of a hybrid example is "Muller *et al.*'s layered InteRRaP architecture shown in Figure 8 developed at the German Research Centre for AI. It is an architecture that implements a layered approach to agent design.

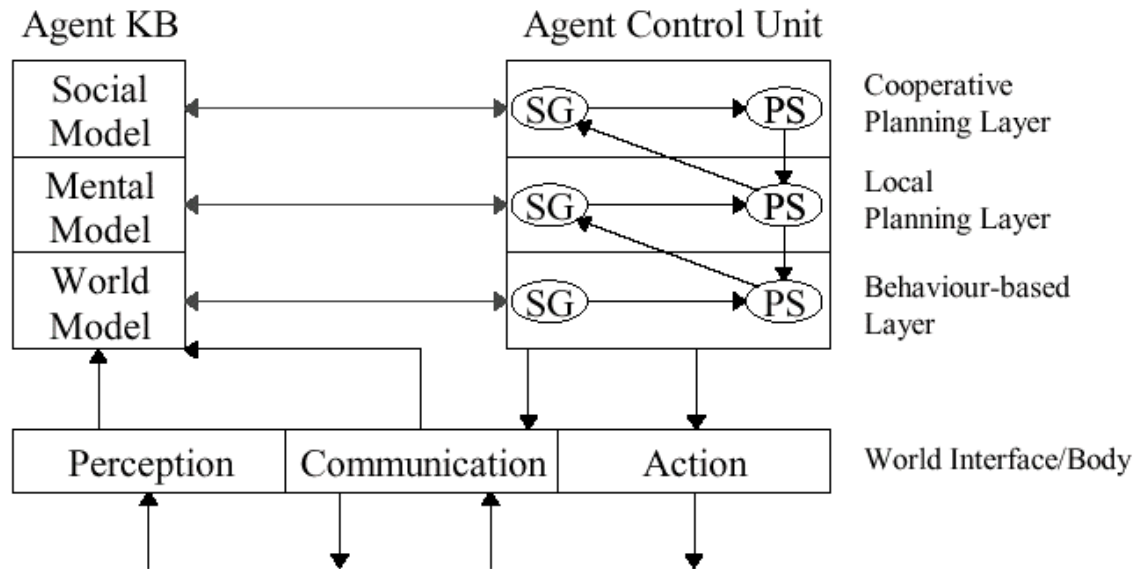


Figure 8 - The InteRRaP Hybrid Architecture ¹⁰

This architecture can be used to construct an agent such as an autonomous robot. As shown, it consists of an agent knowledge base and its associated control unit sitting 'on top' of the perception-action component, which also handles the low-level communications. There are three control layers in this

architecture: the behavior-based layer (BBL), the local planning layer (LPL) and the cooperative planning layer (CPL). " ¹⁰

Heterogeneous Agents

Heterogeneous agent systems refer to "an integrated set-up of at least two or more agents, which belong to two or more different agent classes.

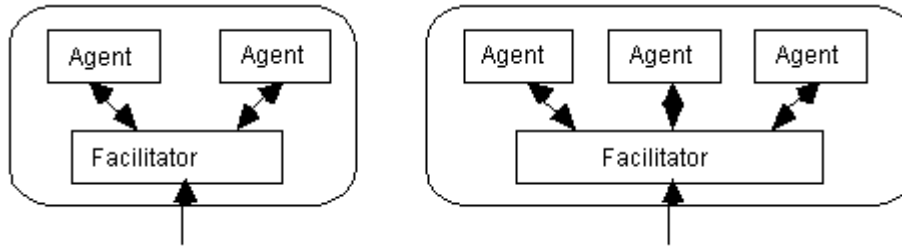


Figure 9 - A Federated System ¹¹

There are five agents distributed in two machines, one with two agents and the other with three. The agents do not communicate directly with one another but do so through intermediaries called facilitators. The agents surrender some of their autonomy to the facilitators who are able to locate other agents on the network capable of providing various services. They also establish the connection across the environments and ensure correct 'conversation' amongst agents." ¹¹

Future of Agents

There are various issues with Intelligent Agents. They include the following:

- Privacy: how do you ensure your agents maintain your much-needed privacy when acting on your behalf?
- Responsibility: when you relinquish some of your responsibility to software agent(s) be aware of the authority that is being transferred to it/them.
- Legal issues: following on from the latter, imagine your agent offers some bad advice to other peer agents resulting in liabilities to other person, who is responsible? The company who wrote the agent?
- Ethical issues: David Eichmann proposed "an agent etiquette for information service and user agents as they gather information on the WWW. They include the following:
 - Agents must identify themselves;
 - They must moderate the pace and frequency of their requests to some server;
 - They must limit their searches to appropriate servers;
 - They must share information with others;
 - They must respect the authority placed on them by server operators;

- An agent's services must be accurate and up-to-date.
- Etzioni & Weld (1994) have proposed others including:
- Safety - the agent should not destructively alter the world;
- Tidiness - the agent should leave the world as it found it;
- Thrift - the agent should limit its consumption of scarce resources;
- Vigilance - the agent should not allow client actions with unanticipated results." ¹²

Distributed Knowledge Based Management System

"A DKBMS is a system that manages the integration of distributed objects into a functioning whole producing, maintaining, and enhancing a business knowledge base. A business knowledge base is the set of data, validated models, metamodels, and software used for manipulating these, pertaining to the enterprise, produced either by using a DKBMS, or imported from other sources upon creation of a DKBMS." ¹³

DKBMS Creation

DKBMSs may be created from scratch, or from a base of varied applications and relatively unintegrated components. DKBMSs are created by integrating already existing components as much as possible, and by adding new components such as data warehouses and data marts where necessary. In addition, an *incremental approach to DKBMS creation is best*, in order to get results more quickly and to build support for the longer term effort to create the DKBMS. ¹³

Intelligent Agents and DKMS

An important aspect of DKBMS architecture is "its Active Knowledge Manager (AKM) component. The AKM provides process control/distribution services, an in-memory active object model accompanied by a persistent object store and connectivity to a variety of data stores and application types. One way the AKM can perform some of its process control services is through software agents. The DKBMS is a system providing support for business processes composed of Planning, Acting, Monitoring, and Evaluating sub-processes. In turn, these sub-processes are composed of more specific sub-processes or use cases, and these, in turn, are composed of tasks. A comprehensive analysis of how agents can participate in process control services, would need to go through the details of how each use case and task sub-process might be performed by agents.

Process Control Services in the DKBMS include:

- in - memory proactive object state management and synchronization across distributed objects;
- component management;
- use case and workflow management; and
- transactional multithreading. " ¹⁴

Object/Component Management and Agents

In the DKBMS, "business objects will be shared across data warehouse and data mart applications, and will be stored in an in-memory object model. The DKBMS through the AKM must have the ability to monitor and coordinate changes in the shared classes and objects across these applications and across their different physical platforms. This means the ability to monitor and coordinate changes in attributes and methods of the shared objects automatically. Let's call this ability Dynamic Integration. To perform dynamic integration, the system must look for changes in shared objects and additions to the total pool of objects and relationships, alert all system components sharing the objects of such changes, and also make decisions about which changes should be implemented in each affected component throughout the system.

It is important that changes in shared objects are propagated and new objects are created in real-time, so that a single view of the DKBMS object model is maintained. This is why in-memory, proactive operation is so important. Like objects, components can also be shared across applications and physical platforms. Component management is the ability to monitor, coordinates, and synchronize changes in components, and is analogous to object state management at the component level. It too, needs to be performed in real-time, and it too requires proactive, in-memory operation to be most effective.

Agents can play a major role in performing dynamic integration as part of the AKM. The AKM is itself composed of distributed object models, made up of reflexive objects. A reflexive object is one that is aware of changes in its state. When a change is introduced in one of these objects it communicates the change (through an alert) to a central object model within the AKM. The central object model contains a view of all objects and relationships in the DKBMS. The central object model will respond to this alert by incorporating the changes into the central object model and deleting the old versions of the objects, as long as no other object models share the old object versions. If they do, the central object model will dispatch Negotiator mobile agents to the various distributed object models incorporating old object versions.

The task of these Negotiator mobile agents is to negotiate with the effected distributed object models about whether the changed objects are acceptable to them. The distributed object models can employ static agents to negotiate for them. If the changed objects are acceptable, the old versions of the objects can be deleted from all object models, and the new objects can be incorporated into all distributed object models. If not, the central object model will maintain both the old and the changed objects to accommodate disagreements among the distributed applications." ¹⁴

Use Case/Workflow Management and Agents

DKBMSs "support business processes by assisting efforts to gather, organize, create, maintain, and enhance knowledge about them, and also by providing support for planning, implementing, monitoring, and evaluating the course of the business process. The use case concept looks at a task sequence from the point of view of the valued outcome the user will get from a task sequence. Workflow, on the other hand, refers to the automated system constructed to implement a use case, a part of a use case, or a set of related use cases.

In deciding whether to store the product of a work flow task or push it to the next step, negotiator agents of the components performing the steps can exchange information on the depth of their work queues; and on their relative abilities to store and process the next step in the work flow. Together they can decide on whether the workflow item in question will be stored or "pushed." In case of disagreement the central AKM component can arbitrate.

Agents based at each component, can also increase the capability to distribute the work flow process by continuously monitoring their components and alerting the central AKM component if processing capability is stressed. The central AKM agent can then assist the "local" agents in negotiations to distribute the workload. Knowledge Retrieval agents, next, can help in providing the capability to gather knowledge resources to support a workflow. Such agents can model each individual information or knowledge resource within the DKBMS. They can then collaborate with Interface agents, receiving queries from them and transmitting only the results to the interface agents. Such agents including descriptive, impact-related, predictive, outcome assessment and benefit/cost assessment knowledge may retrieve various types of knowledge.

Intelligent Interface agents can support collaborative workflow activity in useful ways. For example, in planning, a number of decision-makers may have to agree on a hierarchy of goals and objectives, and ultimately on a planning option. Interface agents can help planners to be explicit about the goals, objectives and priorities that comprise their planning hierarchies. Then negotiating agents for different planners can work together to analyze the similarities and differences in planning hierarchies and to negotiate a common planning hierarchy.

Agents can also assist in simulating workflow systems. Agents functioning as the nodes of a workflow can represent systems. Agents can be assigned tasks they perform according to rules programmed in the agents and triggered by events and their parameters. Workflow items can be defined to provide agents something to process. When the simulation is run various characteristics of the workflow design can be evaluated.

Agents provide only one way to integrate custom, legacy, or external data and applications into a work flow system. But agent technology can be used to produce a

simple information agent by "wrapping" any information source to allow it to conform to the communication conventions of an agent infrastructure." ¹⁴

Tools

Search Agents

[WebSeeker](#) is a meta-searcher. It resides above search engines, such as Yahoo, Lycos, Webcrawler, etc. WebSeeker performs a comprehensive search of numerous different search engines by creating individual keyword searches for each engine. WebSeeker then brings back the results, sorts them, removes duplicates, collates them, and presents the results to the user. The user can then view the results in list form, or as an HTML Web page.

In WebSeeker, you can create a WebSeeker file (.WSK file) containing your search, and the results for the search. A .WSK file can be named after a user, a broad subject, or anything that the user chooses. Each .WSK file contains a keyword search for each search engine loaded in WebSeeker. Each search returns results. All of the results are listed together in the display window. We used the WebSeeker and search for "Intelligent Agents". This is the result of our web search. Click [webseeker.html](#)

Personal Assistance

[TextAloud](#) MP3 lets you listen to text you copy to the clipboard. It uses 'Text to Speech' technology to synthesize natural sounding speech from ordinary text. Just copy text to the clipboard and listen as TextAloud MP3 reads it back to you! TextAloud MP3 also gives you the option of writing text to audio files, so you can listen later. You can create ordinary wav files, or better yet, save the output in compressed MP3 format. Create MP3 files from your email, news articles, and any text you want, and download these files to your portable MP3 player. To test this personal assistance agent, we copied the abstract section of this paper into the clipboard and use read to file function to save the output to a mp3 format. Click [abstract.mp3](#).

References

1. Jeffrey M. Bradshaw, [Introduction to Software Agents](#), in "Software Agents" ,AAAI Press/The MIT Press, 1997.
2. Hyacinth S. Nwana, Software Agents: An Overview, Knowledge Engineering Review, Vol. 11, No 3, pp.1-40, Sept 1996. © Cambridge University Press, 1996
3. Huhns, M. N. & Singh, M. P. (1994), "Distributed Artificial Intelligence for Information Systems", CKBS-94 Tutorial, June 15, University of Keele, UK.
4. Sycara, K. (1995), "Intelligent Agents and the Information Revolution", *UNICOM Seminar on Intelligent Agents and their Business Applications*, 8-9 November, London, 143-159.
5. Finin, T. & Wiederhold, G. (1991), "An Overview of KQML: A Knowledge Query and Manipulation Language", Department of Computer Science, Stanford University.
6. Maes, P. (1994), "Agents that Reduce Work and Information Overload", *Communications of the ACM* 37 (7), 31-40.
7. Wayner, P. (1995), *Agents Unleashed: A Public Domain Look at Agent Technology*, Boston, MA: AP Professional.
8. Indermaur, K. (1995), "Baby Steps", *Byte*, March, 97-104.
9. Brooks, R. A. (1991), "Elephants Don't Play Chess", In Maes, P. (ed) (1991), *Designing Autonomous Agents: Theory and Practice from Biology to Engineering and Back*, London: The MIT press, 3-15.
10. Fisher, K., Muller, J. P. & Pischel, M. (1996), "Unifying Control in a Layered Agent Architecture", *Technical Report TM-94-05*, German Research Center for AI - (DFKI GmbH).
11. Genesereth, M. R. & Ketchpel, S. P. (1994), "Software Agents", *Communications of the ACM* 37 (7), 48-53.
12. Eichmann, D. T. (1994), "Ethical Web Agents", *Proceedings of the 2nd WWW Conference*, <http://www.ncsa.uiuc.edu/SDG/IT94/>
13. Joseph M. Firestone (1997), Distributed Knowledge Management Systems (DKMS): The Next Wave in DSS .D.
14. Joseph M. Firestone (1998), M DKMS Brief No. Three: Software Agents in Distributed Knowledge Management Systems