Agent technologies and their application to Mobile Computing

Lecturer :

Dr. Drake Patrick Mirembe,

E:dpmirembe@gmail.com

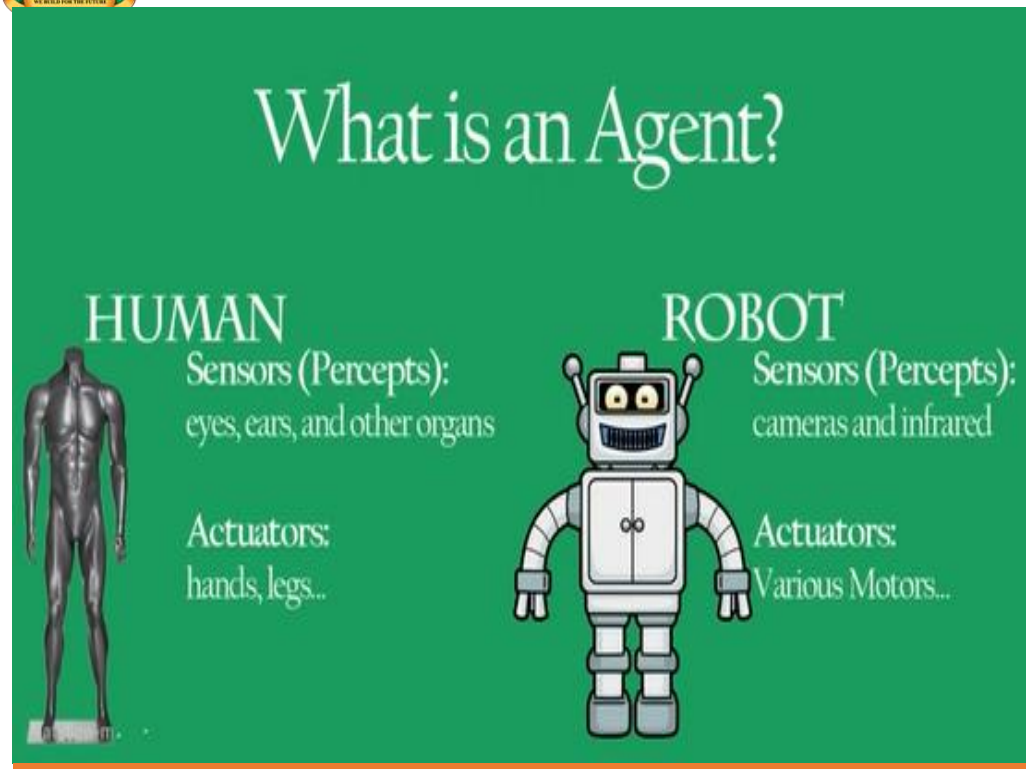
Website: www.drakemirembe.org

Presentation Overview

1. Agents overview
2. Introduction to Agent technologies
3. Mobile agent Technology
4. Software agent Technology
5. KAoS Agent Technology

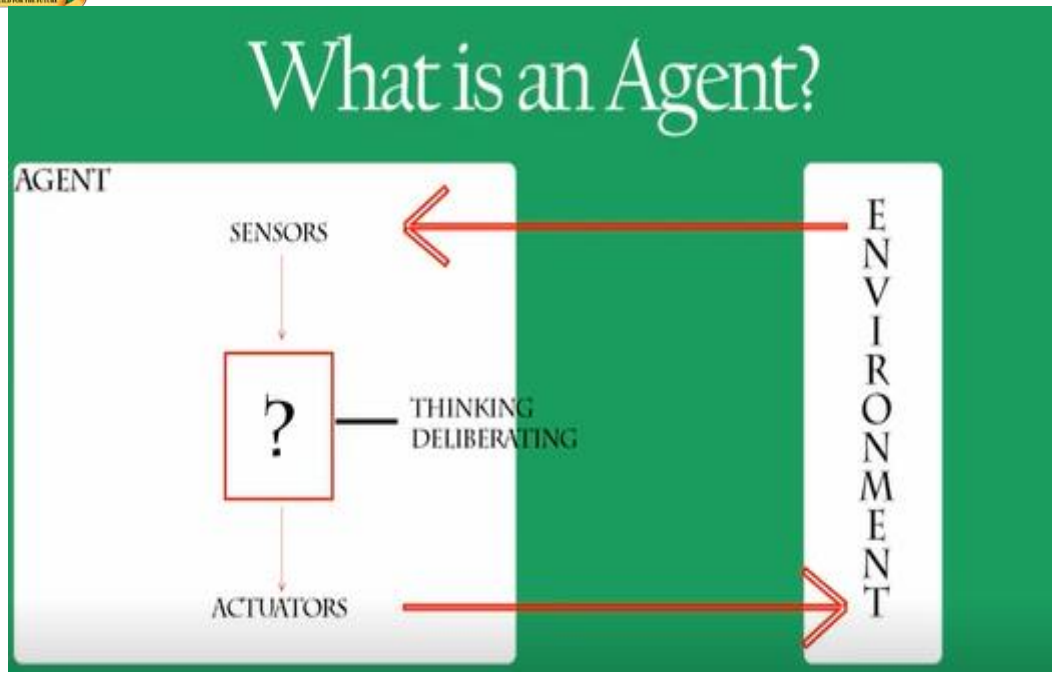
Agents overview

1. Description
2. Characteristics of an agent
3. Classification of agents
4. Applications of agents



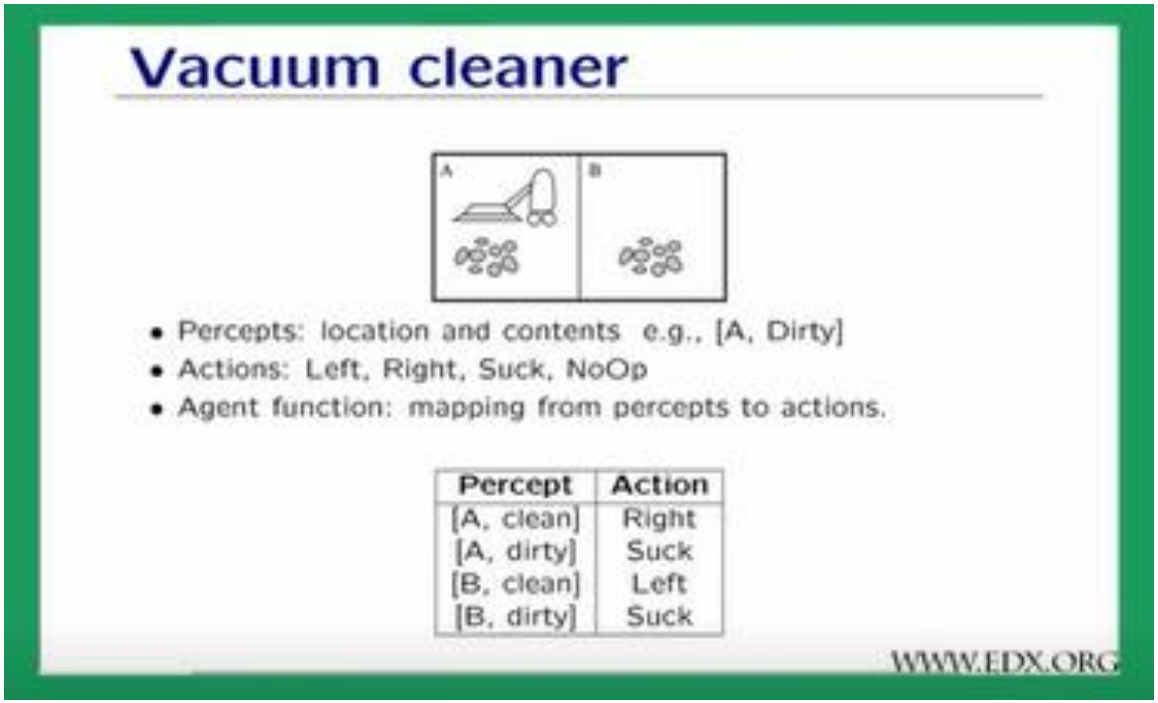
The term *agent is original of the Artificial Intelligence (AI) community, and its notion is* central to the definition of AI itself. A common definition for AI is *'the sub-field of computer science that aims at the construction of agents that exhibit aspects of intelligent behaviour'* (Wooldridge, 1994a).

- An agent is anything that perceives it's environment through sensors and act upon the environment through actuators " — Russell & Norving
- Agent uses various sensors to view an environment and uses actuators to act upon that environment, for-example
- Humans are agents, human agents have eyes, ears & other organs as sensors & hands, legs as actuators.
- Robotic agents may have cameras & Infrareds as sensors and various motors as actuators
- Software agents receive keystrokes, file content & network packets as sensory inputs and acts on that environment by displaying contents on the screen



An Agent runs in cycles;

1. It perceives its environment
2. It thinks about that environment
3. Acts on that environment



- An agent behavior can be described by agents function that maps any given percepts sequence to an action. The map would be stored in a tabular way
- For-example lets look at the Robot vacuum cleaner.
- Vacuum cleaner is perceiving its environment through its senses, thinking or determining if the tile is clean or not, making decisions using its suction as an actuator to act upon that environment by sucking up the dirt. That's what an agent is.

Characteristics of an agent

- ***Autonomy:*** agents should be able to perform the majority of their problem solving tasks without the direct intervention of humans or other agents, and they should have a degree of control over their own actions and their own internal state.
- ***Social ability:*** agents should be able to interact, when they deem appropriate, with other artificial agents and humans in order to complete their own problem solving and to help others with their activities. This requires that agents have, as a minimum, a means by which they can communicate their requirements to others and an internal mechanism for deciding when social interactions are appropriate (both in terms of generating appropriate requests and judging incoming requests).
- ***Responsiveness:*** agents should perceive their environment (which may be the physical world, a user, a collection of agents, the INTERNET, etc.) and respond in a timely fashion to changes which occur in it.
- ***Proactiveness:*** agents should not simply act in response to their environment, they should be able to exhibit opportunistic, goal-directed behaviour and take the initiative where it is appropriate.
- ***adaptability*** - the ability of an agent to modify its behaviour over time in response to changing environmental conditions or an increase in knowledge about its problem solving role
- In additionally, agent may have other features. **Intelligent, knowledge, rationality, mobility etc.**

Classes of an agent

- **COLLABORATIVE AGENT**:-An agent which collaborates with other agents to carry out an intended task. Other agents in this category can be Reactive agents, collaborative
- **INTERFACE AGENT**: An interface agent can be considered as a program that can also affect the objects in a direct manipulation interface, but without explicit instruction from the user
- **INFORMATION AGENT**: Information agents are special kind of so-called intelligent software agents
- **REACTIVE AGENT**: Capable of maintaining an ongoing interaction with the environment, and responding in a timely fashion to changes that occur in it.
- **SMART AGENT**: new forms of software agent that interface with other agents forming an artificial Intelligence system. The acronym" SMART" stands for "System for Managing Agents in Real Time".
- **INTELLIGENT AGENT**: The term intelligent agent can refer to any agent that exhibits some amount of intelligence and there is no requirement that the agent have the ability to work with other agents.
- **COURIER AGENT** which transfers a folder to a specified agent on a specific machine.

Classes of an agent

- **DIFFUSION AGENT**, which executes an agent locally and then creates a clone of itself at every site. Scheduling allows the enforcement of policies that tells when and where an agent is executed .
- This scheduling is implemented by **BROKER AGENT,** which maintains the database of service providers. An Agent needs a given service to always consult a broker to identify which agent provides that service.
- **REAR GUARD AGENT**, it is possible that sites in a computer network will fail. When such a failure occurs, agents at that site will no longer be continuing execution. To solve this problem a REAR GUARD AGENT can be deployed so that the execution can proceed. This agent follows the execution process which moves from one site to another. When an agent fails, this REAR GUARD AGENT launches a new agent. On resumption, this REAR GUARD Agent terminates its own action.

Applications of Agents

1. Personalization in Medicine and HealthCare

- Personalized medicine and healthcare promise prediction, prevention and treatment of illness that is targeted to patients' needs
- Personalized medicine is oriented towards the collecting of information from the patient in order to better tailor his/her needs.
- Rapid increase of use of smartphones and 3G and 4G networks has triggered expanded use of health devices and influenced a lot of different medical aspects like healthcare of aging people. Low-cost sensors have led to their integration into a wide range of wearable devices. So through smartphones and tablets patients can access different health data and monitor their daily activities.
- Healthcare wearables are those wearables for measuring metrics that are assumed to provide an indication of a patient's health and state of wellbeing. Recently there has been the emergence of wearables that are able to monitor detailed clinical metrics, such as blood pressure, heart function, glucose and insulin levels, and medicine intake and so on.
- Agent technologies play important role and can significantly help in developing higher-quality services.

Applications of Agents

2. Electronic Market

- Electronic markets may be considered as hybrid systems, in which people and computers are working together in order to reach certain commercial results.
- Agents may be used in all phases of a market transaction as:

  - **Information** - the main purpose is to search for, find, filter and present information required both by potential customers and potential suppliers.

  - **Trading/matching** - the main purpose is to support the tasks involving negotiations. Agents, on behalf of the buyer, could negotiate with suppliers the terms and conditions of delivery and payment, warranties, and the execution or offering of additional services.

  - **Settlement** - agents could act both by: • providing electronic payment and following the occurrence of derived transaction processes related to the deal as packaging, storage, shipping, insurance, customs clearance,

Applications of Agents

3. Flexible Information systems

- The demand for tools to manage the vast amount of information has come with the explosive growth of network information systems, such as the one that can be found in the World Wide Web (WWW).
- The provision of information is a key prerequisite for a successful distributed information system.
- There is the need to concentrate on what can be done to supply the user with the required information, so that communications problems, location of the data, the amount of data to be manipulated, among other issues do not interfere with an adequate provision.
- Agent technology could play a major role in this context allowing the construction of Flexible Information Systems that can operate autonomously or in cooperation with other systems in order to satisfy the specific goals of users and applications supporting distributed information services over an heterogeneous distributed processing environment
- **Information agents** could be created and launched by the user's computational environment in order to travel through the information network and bring information.

Applications of Agents

4. Mobile Communication and Mobile Computing

- In the light of an increasing number of wired and wireless communication services available for accessing and exchanging information, the vision for future communications, i.e "information anytime, any place in any form", is becoming a reality.
- This new paradigm is already manifesting itself as users travel to many different locations with laptops, PDAs , cellular telephones, and so on.
- Mobile Computing Systems are a combination of the following **three elements** {Mazer, 1995}:
  - Stationary computer and network devices, which we will call the "infrastructure";
  - Mobile computers, of varying physical properties (such weight, size, and power) and computational capabilities
  - Wireline and wireless communications channels of varying bandwidth, cost, reliability, and service properties.

motivations for using mobile agents in this mobile environment {Chess,1995}

- Support to lightweight devices;
- necessity of asynchronous methods of searching for information or transaction services;
- the reduction of overall communication traffic over the diverse communication channels that a mobile device may encounter;
- the ability of the agent to engage in a efficient communication with remote servers;
- the ability of the simple mobile computer to interact with complex applications, without necessarily knowing the remote server capabilities;
- the ability to create "personalized services" for the user, by tailoring the agents that move to the server and respond to the user's requests.

Other Applications of Agents

The number of agent based applications being developed and deployed in real world settings is rapidly increasing. Exemplar systems from the field of information management, one of the fastest growing application areas, include:

- The White House has a system which uses intelligent agents to automatically retrieve information in response to the hundreds of requests that it receives via INTERNET everyday (cited in Houlder, 1994). Agents are used to match keywords in the email received with relevant mailing lists of which they are aware.
- The Commander Exception Monitor (produced by Andersen Consulting and Comshare) uses agents to filter information (cited in Houlder, 1994). This product has been used by Hertz, the car rental company, to analyse pricing structures in the car rental business. By excluding trivial changes, the system reduces the equivalent of 28,000 spreadsheets of information about prices, locations, sizes and type of car into something that can easily be monitored by its pricing executives.
- AT&T's PersonaLink (and the as yet unnamed IBM Intelligent Communication Services system) are both based on agent technology (Reinhardt, 1994). The former uses agents to filter messages and search for information on a network, while the latter is an umbrella for smart message routing.

Introduction to agent technologies

• AI technology was born in 1956. AI technologies are actually very much inspired by the nature(ANN — model of brain, genes and chromosomes etc.).
• AI also has no single definition for intelligence. Some definitions of AI organized into 4 categories, which known as four schools of thoughts
    - Acting humanly/ Acting rationally / Thinking humanly / Acting rationally
• The modern approach to AI is considered as building machines to do right thing(Acting rationally).
• In the future, The gap between man-machine will reduce and lead to man-machine coexistent. In order to enable these development among other AI technologies, Agent technology has done a breakthrough.

Motivation for Agent technology

• When we notice the real world (especially the modern world) which we find very dynamic, interconnected, distributed, uncertain, involves many entities. A system with such features is called a complex systems.
• In a complex system, [1 + 1 > 2]. It means, performances of two entities together are higher than addition of performances of individuals(Team work).
• Complex system has another main feature, that removing a component of a system is does not destroy the system, i.e. still survive; otherwise call a complicated system.
• AI has introduced agent technology (multi agent technology) to model any complex system

Agent technologies in detail

1. Mobile Agent Technology

Mobile agent technology has been promoted as an emerging technology that makes it much easier to design, implement, and maintain distributed systems, including cloud computing and sensor networks. It does not provide an infrastructure for only executing autonomous agents but also migrating them between computers.

What is a Mobile Agent?

• Mobile agents are autonomous, intelligent programs that move through a network, searching for and interacting with services on behalf of user. The agent can suspend its execution, migrate to another machine, and then resume execution on the new machine from the point at which it left off.

• Before the advent of Mobile agents, **the communication between the client and server** is achieved by different approaches such as **message passing**, **Remote Procedure Call (RPC) and Remote Evaluation (REV).**

• **In RPC method,** the procedure resides in the server and client sends a data to the procedure that will be executed there, finally the result is back to the client

• In **REV approach** which is different from the RPC, the procedure itself will be sent and the desired result is returned to the client

Agent technologies in detail

Considering the client /Server model as a driver for mobile agents

- We're examining a scenario where a server provides services to a client
- when a client needs a service, it usually sends a request message to the server as shown in fig. 1 through a communication channel whether it is wired or wireless
- In case, the server does not have resources to satisfy the request made by the client, the client sends request to other server having the needed resource to satisfy the client
- This usually increases the inefficient use of network bandwidth.
- This also increases the network traffic and causes delays due to the involvement of more servers. These factors prohibit the widespread use of this model in a mobile device, because disconnection is frequent in the wireless environment.

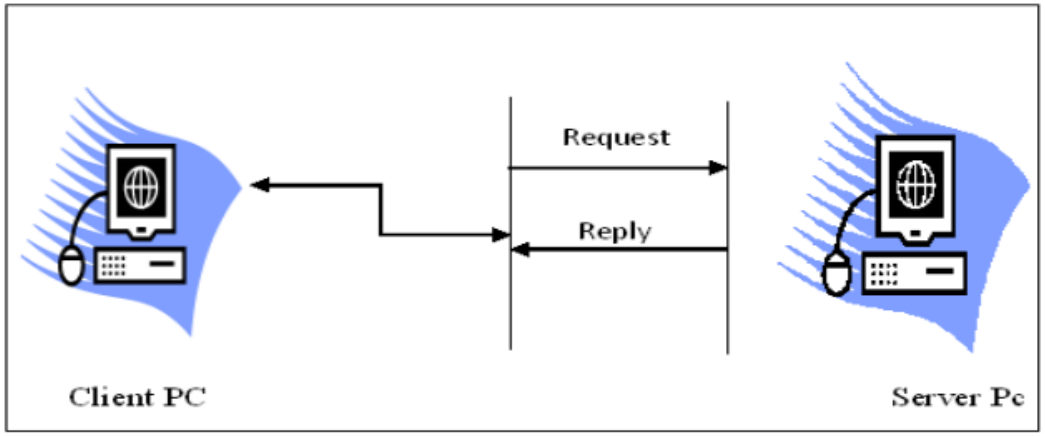


Fig.1 Client/ Server Model

Agent technologies in detail

Considering the client /Server model as a driver for mobile agents

- Mobile agent provides solution for this mobile device because they do not depend on the server operation
- Once the mobile agent has migrated, the connection between the client and server is disconnected, later when mobile agent finishes its job at the server, then it will reconnect to the client or host with the result shown in fig.2.
- This clearly saves the network bandwidth especially in the wireless environment where disconnection is frequent and bandwidth play a major role ("International Journal of Artificial Intelligence & Applications (IJAIA), Vol.3, No.5, September 2012")

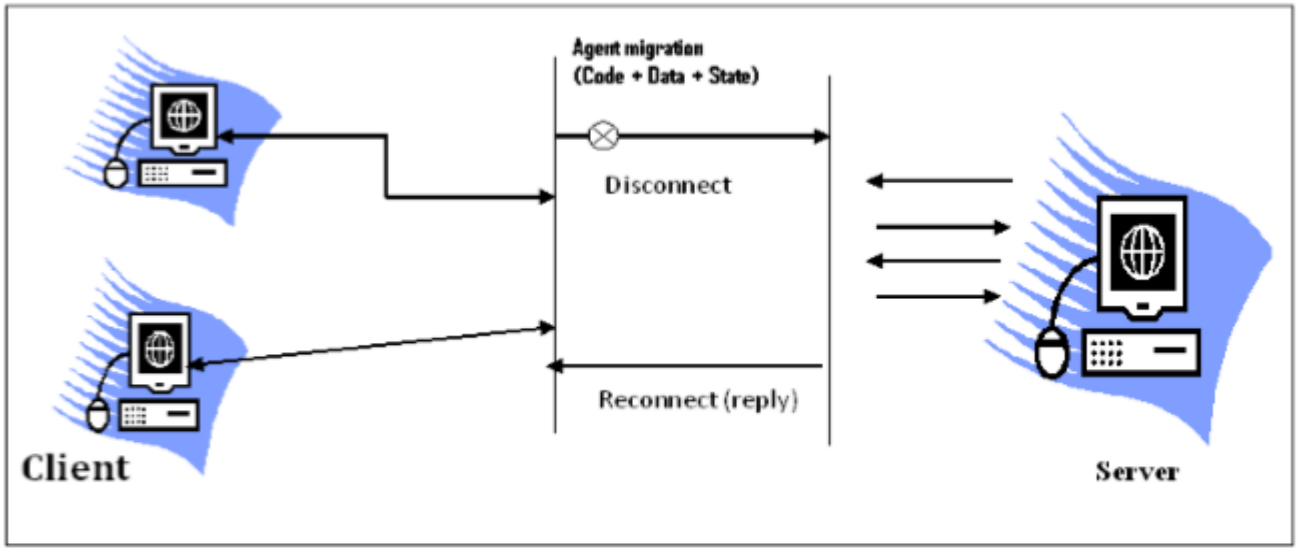


Fig 2 Mobile Agent Model

Agent technologies in detail

Mobile Agent Characteristics

- Autonomous: An agent is able to take initiative and exercise a non-trivial degree of control over its own actions.
- Interactive: means Mobile Agents should communicate with other agents and their environment.
- mobility is the most important property in the Mobile Agent concept, where agent migrated from one node to another within the same environment or in different environment.
- Coordinative; means perform data transfer with other agents in a given environment.
- Proxy: Mobile agents may act on behalf of someone, so they should have certain degree of autonomy.
- Ragged: Mobile Agents should have the ability to deal with the errors whenever occurred.
- Proactive: means they should be goal oriented.
- Cooperative: means coordinate with other agents to achieve a common goal. Mobile Agents should have the capability of learning the current environment and modify its behaviour based on this information.
- Intelligent: means Mobile Agent should be too smart in order to act efficiently.

NOTE: It is not required that agents have all these properties .This is determined by the purpose that agents have to achieve.

Agent technologies in detail

Technical Issue	Implication for Mobile Agents
Bandwidth	MAs conserve bandwidth, especially for networks which have low bandwidth capacity(e.g wireless network). By replacing continous communication with an agent directly at the point of information generation, the bandwidth use can be reduced. Instead of sending dozen or even hundreds of queries across the network, sending one agent on a single request the agent can manage this process locally at the remote side.
Fault-tolerance	MAs can act or respond on errors that may be encountered within their contexts because of their adaptive and ragged attributes.
Flexibility	MAs can give greater flexibility, because new tasks and codes can be added to the system without the need for a fixed code-base.
Interaction	MAs enable new type of interaction, such as negotiating agents that travel to server site seeking for the best deal such as comparing prices(e.g e-commerce application).
Protocols	MAs are able to move to remote hosts in order to establish channels based on proprietary protocols.
Scalability	MAs can carry out their function well (without disruption) when the host system or environment changes in size or volume in order to meet a new user's need.
Self-contained tasks	MAs can carry out tasks which require variable degrees of independence such as network management, software updates etc.
Weak coverage	MAs fit perfectly into a disconnected environment where the signal coverage is frequently lost (being disconnected): MAs will then migrate from one node to another when the coverage becomes available.

Agent technologies in detail

Types of Mobility

1. **Remote execution**
   - the agent is sent before it starts to be executed
   - when it arrives at destination , it is executed until it finishes, agent is transferred once
   - when it is executing, it can use the same remote execution mechanism to start the execution of other agents
   - Destination of the agent is determined by the execution starter
2. **Weak migration**
   - agent can do a weak migration by sending its data along with its code
   - this scheme allows to choose which part of data will be transferred to the new location of the agent.
3. **strong migration**
   - this is the highest degree of mobility
   - using this scheme, not only agent code and data is sent but also the state of execution.
   - when the agent arrives to destination, it is fully restored and its execution is resumed from the same execution point it was just before migration.

Transport of Code and Data	Migration of Code and Data	Migration of Code, Data and State of Execution
→ Remote Execution	→ Weak Migration	→ Strong Migration

\- ——— Mobility degree ——— +

Agent technologies in detail

What happens if the agent itself is big?

- If agent technologies had big agents with lots of code inside, results would not be as good as expected.
- **GH99** is a programming model called **FOAM** (Fragmented Object Agent Model) based on the object model of AspectIX architecture and with a communication layer, **CORBA** (Common Object Request Broker Architecture) compliant, beneath
- Using **FOAM**, an agent doesn't have to be completely transferred during migration but only the required fragment to fulfil the required tasks at the destination agency.

Technological considerations in Mobile agent Technology

Having mobile agents working on mobile devices involves a set of technological components that should provide the expected functionality when all of them work together.

A . Wireless communications

- Mobile devices communicate by using radio signals
- Two most commonly used protocols today for local wireless technologies:
  - Bluetooth
  - Wi-Fi

Agent technologies in detail

- **Bluetooth**
  - was designed for small devices, such as cell phones and PDAs
  - it is normally used for the transmission of small amounts of data or to connect to nearby compatible peripherals (e.g., printers, keyboards, or hands-free headsets).
  - It has a limited bandwidth (maximum 3 Mbps), a range of up to 100 meters, and low power consumption.

- **Wi-Fi networks**
  - based on the IEEE 802.11 standards
  - allow to expand traditional Ethernet local area networks to places where either cabling is not an option or mobility is desired or needed.
  - Its popularity is growing very fast and almost every laptop computer made in the last years has a Wi-Fi interface which allows it to connect to an also increasing number of public access networks in places such as hotels, airports and restaurants.
  - Compared to Bluetooth, it has a higher bandwidth (54 Mbps), a similar range, and a higher power consumption and cost.

Agent technologies in detail

B . Mobile Devices

Mobile devices could be classified in three basic types:

- **Cell phones**. They are small, light, cheap and with little computation capabilities. Data communications can be carried out through mobile phone networks or via bluetooth. Mobile phone networks have a variable bandwidth depending on the transmission technology (4G, 3G, GSM, GPRS, UMTS, etc.) and are available almost everywhere, but using these networks has an economic cost.
- **PDAs or pocket computers.** They are bigger and more expensive than cell phones, but they also have better processing capabilities. There are many architectures (ARM, MIPS, Xscale, etc.) and several operating systems that allow the execution of end user applications similar to those available in desktop computers. Communications can be established through Bluetooth, and more recently also via Wi-Fi.
- **Laptop computers**. They have capabilities comparable to those of desktop computers. They usually have both Ethernet and Wi-Fi interfaces, but it is also possible to use Bluetooth for data interchange with small devices.

Currently, the three types of mobile devices mentioned above are starting to mix.

- cell phones and PDAs are converging into a single device, called **Smartphone** (PDA with a SIM card), being the Apple iPhone one of the most popular Smartphones nowadays.
- laptops and PDAs are mixing into the so-called **Netbooks,** which have less computing capabilities than a conventional PC but also have a lower cost. Another advantage of Netbook is that most of them (like the Asus Eee PC, or the Acer Aspire One) have the samex86 architecture than PCs and can use the same operating systems and applications.

Agent technologies in detail

B . Mobile Devices

Considerations for mobile agent based applications for use in Wireless and Mobile environment

- Features related to security, since the use of wireless communications broadcast data that could be intercepted or altered without having any notice.
- Features related to special network topologies, since there may be multiple mobile nodes with short range and unstable communications, which can make the process of transferring data between two nodes challenging.
- Features related to the way the platform itself works, since the mobile agents need different services, such as transportation or communication services.
- Features that should help the developer of agent-based systems, such as monitoring or debugging tools.

Security in Mobile agent Technology

Most security issues in mobile agents are common to existing computer security problems in communication and the downloading of software**.**

There are **two problems** in mobile agent security:

1. the protection of hosts from malicious mobile agents
2. the protection of mobile agents from malicious hosts.

Agent technologies in detail

Security in Mobile agent Technology

- It is difficult to verify with complete certainty whether an incoming agent is malicious or not. However, there are two solutions to protecting hosts from malicious mobile agents.

  1. The first is to provide access-control mechanisms, e.g., Java's security manager. They explicitly specify the permission of agents and restrict any agent behaviours that are beyond their permissions.

  2. To provide authentication mechanisms by using digital signatures or authentication systems. They explicitly permit run time systems to only receive agents that have been authenticated, have been sent from authenticated computers, or that have originated from authenticated computers

- There have been no general solutions to these second problem, because it is impossible to keep agent private from runtime systems executing the agent. However, (non -malicious) runtime systems can authenticate the destinations of their agents, to check whether these are non-malicious, before they migrate the agents to these destinations

Agent technologies in detail

Applications for Mobile Agent Technology

1. **Remote information retrieval** - agents can migrate among multiple database servers to retrieve and gather the interesting data from the servers. They can also determine the destinations based on information they have acquired from the database servers that they have thus far visited.
2. **Network management** – Mobile agent-based network management has several advantages in comparison with traditional approaches, such as the client/server one.
   - As code is very often smaller than the data it processes, the transmission of mobile agents to sources of data creates less traffic than transferring the data itself. Deploying a mobile agent close to the network nodes that we want to monitor and control prevents delays caused by network congestion.
   - Since a mobile agent is locally executed on the node it is visiting, it can easily access the functions of devices on this node.
   - The dynamic deployment and configuration of new or existing functionalities into a network system are extremely important tasks, especially as they potentially allow out dated systems to be updated in an efficient manner.
   - Network management systems must often handle networks that may have various malfunctions and disconnections and whose exact topology may not be known. Since mobile agents are autonomous entities, they may be able to detect proper destinations or routings on such networks.

Agent technologies in detail

Applications for Mobile Agent Technology

3. **Cloud computing** – In a distributed system, e.g. a grid or cloud computing system, computers tend to be numerous and their computational loads are different. Since mobile agents can migrate to other computers, tasks that are implemented as mobile agents can be relocated at suitable computers whose processors can execute the tasks. This is practical in implementing massively multi agent systems that must operate a huge number of agents, which tend to be dynamically created or which terminate on a distributed system that consists of heterogeneous computers

4. **Mobile computing -** Mobile agents use the capabilities and resources of remote servers to process their tasks. When a user wants to do tasks beyond the capabilities of his or her computers, the agents that perform the tasks can migrate to and be executed at a remote server.

5. **Software testing**

Mobile agents are useful in the development of software as well as the operation of software in distributed and mobile computing settings. An example of these applications is testing methodology for software running on mobile computers, called *Flying Emulator. Wireless LANs* or 4G-networks incorporate wireless LAN technologies, and mobile terminals can access the services provided by LANs, as well as global network services.

Agent technologies in detail

Applications for Mobile Agent Technology

6. Active networking

There are two approaches to implementing active networks. The active packet approach replaces destination addresses in the packets of existing architectures with miniature programs that are interpreted at nodes on arrival. The active node approach enables new protocols to be dynamically deployed at intermediate and end nodes using mobile code techniques. Mobile agents are very similar to active networks, because a mobile agent can be regarded as a specific type of active packet, and an agent platform in traditional networks can be regarded as a specific type of active node

7. Active Documents

Mobile code technology is widely used in plug-in modules for rich internet applications (RIA) in web-browsers, e.g., Java Applet and Macromedia Flash. Such modules provide us with interactive user experiences because their virtual machines, e.g., Java virtual machines and Flash players, can locally execute and render them across multiple platforms and browsers without having to communicate with remote servers

Mobile agent-based modules for RIA can naturally carry both their code and state at client computers. For example, MobiDoc is a mobile agent-based framework for building mobile compound documents where a compound document can be dynamically composed of mobile agent-based components, which view or edit their contents, e.g., text, images, and movies. It can migrate itself over a network as a whole, with all its embedded components.

.

Agent technologies in detail

Applications for Mobile Agent Technology

8. AMBIENT COMPUTING - Ambient computing is one of the most important applications of mobile agents and mobile agents is useful in building and operating ambient computing environments. Ambient computing environments, which consist of computers often have limited resources, such as restricted levels of CPU power and amounts of memory. Mobile agents can help to conserve these limited resources, since each agent only needs to be present at the computer when the computer needs the services provided by that agent.

One of the most typical application of mobile agent technology in ambient computing is "follow-me" application, which tracks the current location of the user and allows him/her to access his/her applications at the nearest computer as he/she moves around in the building.

Agent technologies in detail

Challenges of mobile Agents include the following:

· **Transportation**: how does an agent move from place to place? How does it pack up and move?

· **Authentication**: how do you ensure the agent is who it says it is, and that it is representing who it claims to be representing? How do you know it has navigated various networks without being infected by a virus?

· **Secrecy**: how do you ensure that your agents maintain your privacy? How do you ensure someone else does not read your personal agent and execute it for his own gains? How do you ensure your agent is not killed and its contents „core-dumped"?

· **Security:** how do you protect against viruses? How do you prevent an incoming agent from entering an endless loop and consuming all the CPU cycles?

· **Cash**: how will the agent pay for services? How do you ensure that it does not run amok and run up an outrageous bill on your behalf? In addition to these are the following:

· **Performance issues:** what would be the effect of having hundreds, thousands or millions of such agents on a WAN?

· **Interoperability/communication/brokering services**: how do you provide brokering/directory type services for locating engines and/or specific services? How do you execute an agent written in one agent language on an agent engine written in another language? How do you publish or subscribe to services, or support broadcasting necessary for some other coordination approaches?

Take home for Mobile agent technology

- Mobile agent technology provides a new way of communication over heterogonous network environment. A number of advantages have been proposed and identified which includes:
- efficiency and reduction of network traffic, asynchronous autonomous interaction, interaction with real-time entities, local processing of data, support for heterogeneous environment and having robust and fault-tolerant behaviour.
- However, the security, infrastructure and standardising issues still represent significant constraints.
- mobile agent technology has the potential in increasing the performance of networks as well as for software adopting mobile agents. Due to its nature of being a futuristic technology from the programming environment perspective a lot of work is still required before the average programmer can build applications based on the mobile agent technology paradigm with ease.

Agent technologies in detail

2. SOFTWARE AGENT TECHNOLOGIES

Introduction

Since the 1980's, the metaphor of "agents" has been used to create and handle software products in a variety of application domains. Although the "killer" application is still missing, software agents have undergone a remarkable development. Agent concepts improve the way in which software is developed, and extend the range of applications by making solutions for problems feasible that were barely solvable using traditional technologies.

What is a software agent?

Software agents can be generally defined as entities that function continuously and autonomously in a particular environment that is often inhabited by other agents and processes.

Types of software agents

Agent can be classified in many ways according to service offers.

1. Agent could be **private or public**. For-example personal assistants are private agents and Facebook, amazon, eBay are public agents.
2. Agent could be **stationary or mobile** also, depending on/from where it operates. Mobility is required to identify the best resource location to perform it's roles.

Agent technologies in detail

2. SOFTWARE AGENT TECHNOLOGIES

Value of Software Agents in a networked World

- major value of employing software agents with intranet, Internet, and extranet applications is that they are able to assist in locating and filtering data.
- They save time by making decisions about what is relevant to the user thus reducing the effort required to locate and retrieve data
- They are able to sort through the network and the various databases effortlessly and with unswerving attention to detail in order to extract the best data.
- They are not limited to hard (quantitative) data; they can also obtain soft data about new trends that may cause unanticipated changes (and opportunities) in local or even global markets.
- With an agent at work, the competent user's decision-making ability is enhanced with information rather than paralyzed by too much input.
- improve productivity by off-loading a variety of mundane, tedious, and mindless tasks.

Agent technologies in detail

Motivations / drivers for Software Agent technology

- **Mundane personal activity.** In a fast-paced society, time-strapped people need new ways to minimize the time spent on routine personal tasks such as shopping for groceries or travel planning, so that they can devote more time to professional activities.
- **Search and retrieval**. It is not possible to directly manipulate a distributed database system containing millions of data objects. Users will have to relegate the task of searching and cost comparison to agents. These agents will perform the tedious, time-consuming, and repetitive tasks of searching databases, retrieving and filtering information, and delivering it back to the user.
- **Repetitive office activity.** There is a pressing need to automate tasks performed by administrative and clerical personnel in functions such as sales or customer support in order to reduce labour costs and increase office productivity. Today, labour costs are estimated to be as much as 60 percent of the total cost of information delivery (Abushar and Hirata 2002).
- **Decision support.** There is a need for increased support for tasks performed by knowledge workers, especially in the decision-making area. Timely and knowledgeable decisions made by these professionals greatly increase their effectiveness and the success of their businesses in the marketplace.
- **Domain experts.** It is advisable to model costly expertise and make it widely available. Expert software agents could model real-world agents such as translators, lawyers, diplomats, union negotiators, stockbrokers, and even clergy.

EXHIBIT D.1 Software Agents vs. Traditional Software Programs

Characteristics	Regular Software	Agents
Nature	Static	Dynamic
Manipulation	Direct: User initiates every action	Indirect: Autonomous
Interactivity	Noninteractive	Dialogues are fully interactive. Actions may be initiated by either the user or the agent system. Interacts with user and with other agents.
Flexibility	Never changes, unless changed by a human or an error in the program	Adapts, learns
Temporal continuity	Runs one time, then stops to be run again when called	Persistent: Continues to run over time
Response	Predictable: Does what you tell it to, even if you didn't mean what you said	Interprets what you mean, not what you say. In the best of circumstances, actions are based on rules, but they may change over time or in reaction to different circumstances.
Autonomy, independence	Follows instructions	May initiate actions, as well as respond to instructions
Mobility	Stays in one place	May be mobile, traveling to other servers
Concurrency	Generates process in one dedicated server with limited processing power	Dispatches simultaneously to accomplish various parts of a task in parallel
Local interaction	Accesses data across network using client-server architecture	Can travel and interact with local entities, such as databases, file servers and stationary agent, through message passing

Source: Based on Feldman and Yu (1999).

Agent technologies in detail

INTELLIGENCE LEVELS;

Definitions of agents are greatly dependent on the agents' levels of intelligence, which are described by Lee et al. (1997) as follows:

- **Level 0 (the lowest).** These agents retrieve documents for a user under straight orders. Popular Web browsers fall into this category. The user must specify the URLs where the documents are. These agents help in navigating the Web.

- **Level 1.** These agents provide a user-initiated searching facility for finding relevant Web pages. Internet search agents such as Google, Alta Vista, and Lycos are examples. Information about pages, titles, and word frequency is stored and indexed. When the user provides key words, the search engine matches them against the indexed information. These agents are referred to as *search engines.*

- **Level 2.** These agents maintain users' profiles. They then monitor the Internet and notify the users whenever relevant information is found. An example of such an agent is WebWatcher. Agents at this level are frequently referred to as *semi-intelligent or software agents.*

- **Level 3.** Agents at this level have a learning and deductive component of user profiles to help a user who cannot formalize a query or specify a target for a search. DiffAgent (CMU) and Letizia (MIT) are examples of such agents. At this level are referred to as *learning or truly intelligent agents.*

Introduction to agent technologies

CHARACTERISTICS OF SOFTWARE AGENTS

1. autonomy
2. temporal continuity
3. Reactivity
4. goal driven
5. software agent may also possess additional traits such as adaptability, mobility, sociability, and personality.

1. Autonomy

A software agent senses its environment and acts autonomously upon it. A software agent can initiate communication, monitor events, and perform tasks without the direct intervention of humans or others displaying the following characteristics:

- **Goal-oriented.** Accepts high-level requests indicating what a human wants and is responsible for deciding how and where to satisfy the requests. These are referred to by " Hess et al. (2000)" as *homeostatic goal(s).*
- **Collaborative.** Does not blindly obey commands but can modify requests, ask clarification questions, or even refuse to satisfy certain requests.
- **Flexible.** Actions are not scripted; the agent is able to dynamically choose which actions to invoke, and in what sequence, in response to the state of its external environment.
- **Self-starting.** Unlike standard programs directly invoked by a user, an agent can sense changes in its environment and decide when to act.

Introduction to agent technologies

CHARACTERISTICS OF SOFTWARE AGENTS

2. Temporal continuity - A software agent is a program to which a user assigns a goal or task. The idea is that once a task or goal has been delegated, it is up to the agent to work tirelessly in pursuit of that goal. Unlike regular computer programs that terminate when processing is complete, an agent continues to run—either actively in the foreground or sleeping in the background—monitoring system events that trigger its actions. You can think of this attribute as "set and forget."

3. Reactivity - A software agent responds in a timely fashion to changes in its environment. This characteristic is crucial for delegation and automation. The general principle on which software agents operate is "When X happens, do Y," where X is some system or network event that the agent continually monitors (Gilbert 1997).

4. GOAL DRIVEN - A software agent does more than simply respond to changes in its environment. An agent can accept high-level requests specifying the goals of a human user (or another agent) and decide how and where to satisfy the requests. In some cases, an agent can modify the goals or establish goals of their own.

5. COMMUNICATION (INTERACTIVITY) -

Many agents are designed to interact with other agents, humans, or software programs. Instead of making a single agent conduct several tasks, additional agents can be created to handle un delegated tasks. Thus, communication is necessary. Agents communicate by following certain communication languages and standards such as ACL and KQML (Bradshaw 1997; Jennings et al. 1998).

Agent technologies in detail

6. Intelligence and learning

Currently, the majority of agents are not truly intelligent because they cannot learn; only some agents can learn. This goes beyond mere rule-based reasoning because the agent is expected to use learning to behave autonomously. Although many in the AI community argue that few people want agents who learn by "spying" on their users, the ability to learn often begins with the ability to observe users and predict their behaviour. One of the most common examples of learning agents is the wizards found in many commercial software programs (e.g., in Microsoft Office applications). These wizards offer hints to the user, based on patterns the program detects in the user's activities.

SIMPLE SOFTWARE AGENTS

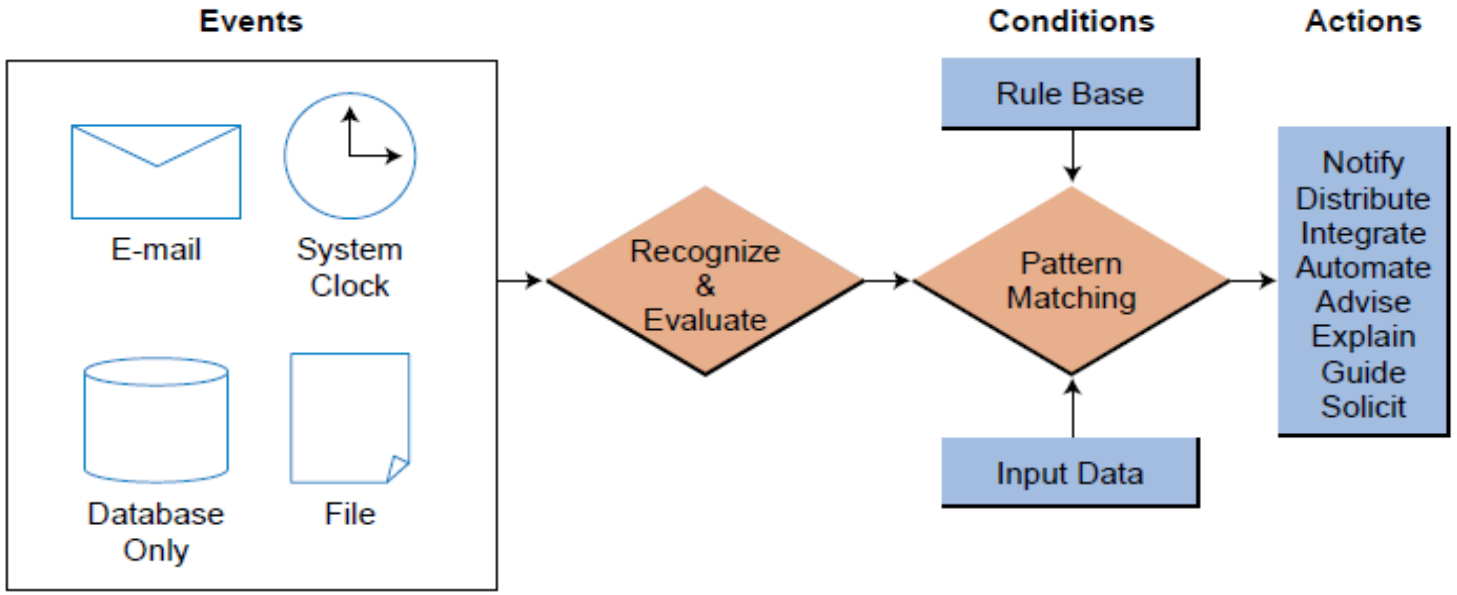
- Simple agents work within the context of a single application and focus on a single set of tasks with a circumscribed set of outcomes.
- These agents automates simple repetitive tasks that could be performed by a person, if that person had the time, the inclination, or was available to do so. This is certainly the case with e-mail agents.
- E-mail agents sole purpose is to scan incoming and outgoing messages, looking for various keywords that have been designated by the end user and performing one or more of a handful of possible operations, such as deleting the message, forwarding the message, or storing the message within a given folder. For example, an agent **at hotmail.com** decides which mail to place in the "junk mail" file. It will also block any mail you ask to be blocked.

Agent technologies in detail

- e-mail agents have their advantages. They never sleep (*unless the application or the system is shut down),* They send you an automatic reply: "Professor Turban is away until July 10."
- They are always available, even when the end user is away from their desk.
- They are never bored, and they never miss work. Some executives, managers, and knowledge workers receive more than 100 to 200 messages a day, and reviewing these messages can be a tedious, time-consuming, and error-prone task. Automating the review with an e-mail agent can off-load some of the review process.
- The goals of a simple software agent are explicitly specified by an end user.
- This is done by either creating a set of "if/then/else" rules or a script that predefines the actions to be taken by the agent when certain conditions arise. The actions are invoked by the agent without end-user intervention.

Agent technologies in detail

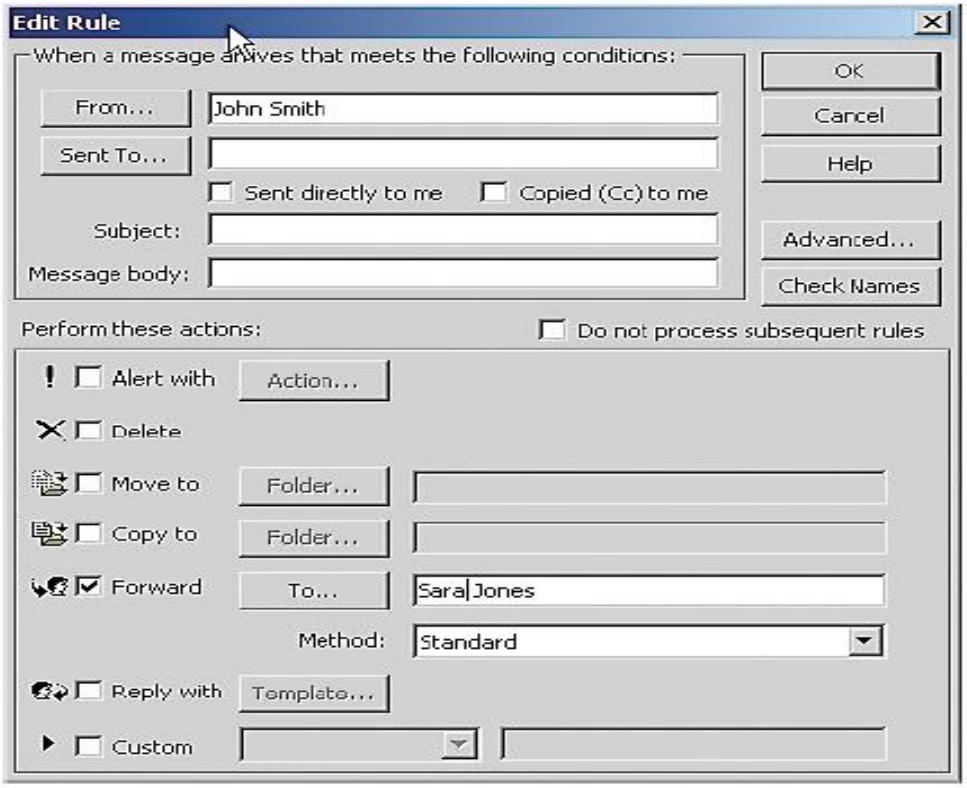
EXHIBIT D.3 Operation of a Simple Agent



In the case of an e-mail agent, the agent simply compares an incoming or outgoing message against all of the end user's if/then rules. If a message satisfies the conditions specified in the "if " part of the rule, the actions designated in the "then" part are carried out by the agent. By 2002, popular e-mail programs such as Outlook Express, Eudora, and Netscape Mail included dozens of agents for e-mail creation and management, such as auto respond, new mail notification, and spell checking.

Agent technologies in detail

EXHIBIT D.4 Out-of-Office Agent in Microsoft E-Mail



the rule would read something like, "When the 'out-of-the-office' switch is on, if a message arrives that has the exact words 'John Smith' in the 'From' field, then 'Forward' the message to 'Sarah Jones.'"

Agent technologies in detail

LEARNING AGENTS

❖Software agents are called intelligent agents or **learning agents** if they have the capacity to adapt or modify their behaviour; that is, to learn.
❖Simple software agents, such as e-mail agents, lack this capacity. If a simple software agent has any intelligence at all, it is found in the subroutines or methods that the agent uses to do pattern matching. However, these subroutines or methods are built into the program and cannot be modified by the agent.

As Maes suggests, there are four ways for an interface agent to modify its behaviour:
1. **"Look over the shoulder" of the user.** An agent can continually monitor the user's interactions with the computer. By keeping track of the user's actions over an extended period of time, the agent can discern regularities or recurrent patterns and offer to automate these patterns.
2. **Direct and indirect user feedback.** The user can provide the agent with negative feedback either in a direct or indirect fashion. Directly, the user can tell the agent not to repeat a particular action. Indirectly, the user can neglect the advice offered by an agent and take a different course of action.
3. **Learn from examples given by the user.** The user can train the agent by providing it with hypothetical examples of events and actions that indicate how the agent should behave in similar situations.
4. **Ask the agents of other users.** If an agent encounters a situation for which it has no recommended plan of action, it can ask other agents what actions they would recommend for that situation.

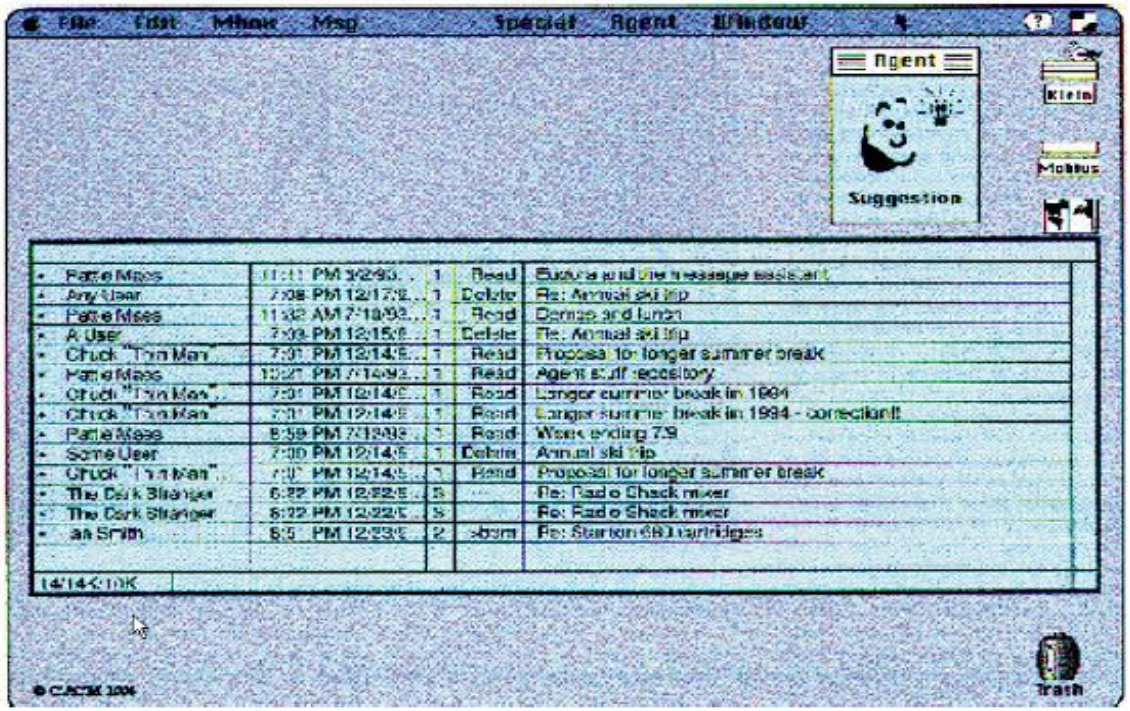
Agent technologies in detail

LEARNING AGENTS

- The major difference between the operation of an intelligent learning agent and the workings of a simple software agent is in how the if/then rules are created.
- With a learning agent, the onus of creating and managing rules rests on the shoulders of the agent, not the end user.
- Let's consider Maxim (Maes 1994) intelligent e-mail agent that operates on top of the Eudora e-mail system that relies on a form of learning known as *case-based reasoning.*
- Maxim continually monitors what the user does and stores this information as examples.
- The situations are described in terms of fields and keywords in the message (i.e., the "From," "To," and "Cc" lists, the keywords in the "Subject" field, and so on), and the actions are those performed by the user with respect to the message (e.g., the order in which the user reads it, whether the user deleted or stored it, and so on).
- When a new situation occurs, the agent analyzes its features based on its stored cases and suggests an action to the user (such as read, delete, forward, or archive).
- The agent measures the *confidence, or fit, of a suggested action to a situation.*
- Two levels of confidence are used to determine what the agent actually does with its suggestion; - If the confidence is above the "do-it" threshold, the agent automatically executes the suggestion. - If the confidence is above the "tell-me" threshold, the agent will offer a suggestion and wait for input from the user

Agent technologies in detail

EXHIBIT D.5 Intelligent E-Mail Agent



Source: This intelligent e-mail agent is from software called Maxim, written at MIT. For more details, see Maes 1994.

critics argue that most people do not want intelligent agents "looking over their shoulders" (Greif 1994). They contend that simple software agents that require end users to fill out forms are easy to use and provide enough utility for the average end user.

Agent technologies in detail

MULTI-AGENTS AND COMMUNITIES OF AGENTS

❑ In executing complex tasks that require much knowledge, it is necessary to employ several software agents in one application. These agents need to share their knowledge, or the results of applying this knowledge together may fail.

❑ An example is routing among telecommunications networks. Information can pass through a network controlled by one company into another network controlled by another company.

❑ Computers that control a telecommunications network might find it beneficial to enter into agreements with other computers that control other networks about routing packets more efficiently from source to destination.

❑ Another example is wireless devices that are continuously increasing their functionality. Wireless devices offer more than Internet access and e-commerce support, they also enable device-to-device communication. You can take a photograph with your digital camera in one location and transmit pictures wirelessly, in seconds, to your office. Intelligent agents embedded in such devices facilitate this interaction.

Agent technologies in detail

MULTI-AGENT SYSTEMS

• In multi-agent systems, there is no single designer who stands behind all the agents.

• Each agent in the system may be working toward different goals, even contradictory ones. Agents either compete or cooperate (Decker et al. 1999).

• In a multi-agent system, for example, a customer may want to place a long-distance call. Once this information is known, agents representing the carriers submit bids simultaneously. The bids are collected, and the best bid wins.

• In a complex system, the customer's agent may take the process one step further by showing all bidders the offers, allowing them to rebid or negotiate.

• A complex task is broken into subtasks, each of which is assigned to an agent that works on its task independently of others and is supported by a knowledge base.

• Acquiring and interpreting information is done by knowledge-processing agents that use deductive and inductive methods, as well as computations.

• The data is refined, interpreted, and sent to the coordinator, who transfers to the user interface whatever is relevant to a specific user's inquiry or need. If no existing knowledge is available to answer an inquiry, knowledge creating and collecting agents of various types are triggered.

Agent technologies in detail

Important Applications from a Historical Point of View
The earliest applications of software agents can be found in the early 1980's. The first publications on agent-related technologies like the contract net protocol [49], or blackboard systems originate from this time.

➢**The Distributed Vehicle Monitoring Testbed (DVMT) -** is one of the first, and at its time, most influential multi-agent systems. Agents have to monitor and interpret data from spatially distributed sensors. The overall task of the system was to determine a global picture of vehicle traffic across the complete sensing area. Every agent is responsible for a certain part of the complete area.

➢ The **YAMS system -** was another influential project using the contract net protocol for manufacturing control. It was one of the first industrial applications of multi-agent systems.

➢ The largest project in Europe was **the ARCHON** (Architecture for Cooperative Heterogeneous ON-line systems) project [26] 1989-1993. Practical applications of the developed architecture and framework were in process control, e.g. the CERN particle accelerator, or electricity transportation management at Iberdrola. The multi-agent systems partially consisted of legacy systems which received an agent wrapper that enables interaction and cooperation capabilities.

agent technologies explained

Important Applications from a Historical Point of View

➢ At the beginning of the 90's, At the Australian Artificial Intelligence Institute, **the OASIS system (Optimal Aircraft Sequencing using Intelligent Scheduling)** was developed for air traffic control. It has actually been applied in a field study at the airport of Sydney. When an aircraft enters the airspace of Sydney, an agent is assigned to it. Information and goals of these agents are set according to the flights goal and the information that the plane might have. This agent then negotiates with air traffic management agents for landing time and runaway allocation, or with other agents for conflict free paths.

➢ Also in the area of personal assistants, the first successes were at the beginning of the 1990's. Maes presents a set of different agents that support users in mail management, or Internet news filtering which all build on the related technologies.

agent technologies explained

Current Applications

❑ Industrial applications in production planning and scheduling, as well as in real-time control and supply chain management where software agents can support complex coordination of the participants

❑ Another important application is workflow management. The ADEPT project (advanced decision environment for process tasks) shows how business process management can be accomplished using negotiating agents

❑Agent applications in the telecommunication sector have been being developed for more than ten years. They have provided support for telecommunication services, or service management, like the ADEPT project. Agent-based routing in telecommunication networks is mostly associated with simple agents with behaviour inspired by the emergence of ant-trails.

❑ Health care and knowledge management

❑ A relatively new application domain is bioinformatics where agent-based systems are expected to be able to cope with the immense volume of DNA data and the complex tasks that have to be solved.

❑ traffic and transportation management

❑ Air traffic control

❑ Defence

❑Shopping agents are ideal applications of Agent Builder agents. These agents can be used to locate merchandise, compare prices, place orders

❑ The most favourite area with respect to reactive agents is games and entertainment industry.

agent technologies explained

Prospects for Software Agents

In foreseeing the future of information technology, Grover and Segars emphasized four important trends;

1. hardware is evolving in the direction of smaller and more powerful machines;
2. software is evolving in the direction of user-friendly, modular and flexible platforms;
3. cost of digitizing is approaching zero - Cost of digitizing approaching zero means that every single piece of information interpreted by our brain through our five senses can be represented in digital form at no cost.
4. cost of coordinating, which refers to all costs involved in acquiring and evaluating information, is also approaching zero - The cost of coordinating approaching zero means that there will be a lot of opportunity for innovation in companies and for securing better services by customers.

➢ Trends in artificial intelligence applications - Artificial Intelligence (AI) is concerned with programs that respond flexibly in situations that were not specifically anticipated by the programmer. To make computers/programs smarter and smarter and come closer and closer to imitating, or ultimately surpassing, complex human thought processes, we need to draw upon advances in artificial intelligence.

➢ Robotics has always been a branch of AI research and application. Over the years, Robots have become apparent that in addition to never getting bored, never getting sick and never going on strike, robots can be designed to work far more rapidly and precisely than humans alone.

agent technologies explained

General Issues of Software Agents

❑ **Privacy:** how do you ensure your agents maintain your much needed privacy when acting on your behalf?

❑ **Responsibility** which goes with relinquished authority: when you relinquish some of your responsibility to software agent(s), be aware of the authority that is being transferred to it/them. How would you like to come back home after a long hard day at work being the proud owner of a used car negotiated and bought for, courtesy of one of your software agents? How do you ensure the agent does not run up a huge credit card bill on your behalf?

❑ **Legal issues**: following on from the latter, imagine your agent offers some bad advice to other peer agents resulting in liabilities to other people, who is responsible? The company who wrote the agent? You who customized it? or Both? We envisage a new raft of legislation would need to be developed in the future to cover software agents.

❑**Ethical issues:** these would also need to be considered. Already, (Norman) is already concerned enough about the ethics of software agents that he has proposed an agent etiquette for information service and user agents as they gather information on the WWW. .

CONCLUSION ON SOFTWARE AGENT TECHNOLOGY

Software agents will not be all pervasive. They will not magically solve all the difficult problems which exist in the current generation of advanced information processing systems e.g planning in uncertain environments, perceiving and acting in a timely fashion in response to environmental changes, and inferring a user's preferences based on their behaviour. Moreover, by their very nature software agents create a new set of problems which must be tackled. Because they are autonomous, users may be wary in trusting them to act on their behalf

3. KAoS AGENT TECHNOLOGY

" https://www.researchgate.net/publication/2814077"

Author – Jeffrey M. Bradshaw

History

KAoS grows out of work beginning in 1988 on a general purpose inter-application communication mechanism for the Macintosh called MANIAC (Manager for Inter-Application Communication) (Bradshaw et al. 1991, 1988). Plans for coordination among MANIAC-enabled applications were modelled and executed by means of an integrated planner developed using ParcPlace Smalltalk. A later version, NetMANIAC, extended messaging capabilities to other platforms through the use of TCP/IP. In 1992, a collaboration with the Seattle University (SU) Software Engineering program to develop the first version of KAoS (Tockey et al. 1995; George et al. 1994), they replaced the integrated planner with a fully object-oriented agent framework, borrowing ideas from Shoham's (1997) AGENT-0 work. The following year, a new group of students replaced the MANIAC capability with HP Distributed Smalltalk's version of OMG's Common Object RequestBroker Architecture (CORBA) (Siegel 1996).

3. KAoS AGENT TECHNOLOGY

Motivation

The long-term objective of the KAoS (Knowledgeable Agent-oriented System) agent architecture is to address **two major limitations** of current agent technology:

1. Failure to address infrastructure, scalability, and security issues; and
2. lack of semantics and extensibility of agent communication languages.

Suggested solutions;

- ❑ The first problem is addressed by taking advantage of the capabilities of commercial distributed object products (CORBA, DCOM, Java) as a foundation for agent functionality, and supporting collaborative research and standards-based efforts to resolve agent interoperability issues.
- ❑ The second problem is addressed by providing an open agent communication meta-architecture in which any number of agent communication languages with their accompanying semantics could be accommodated. Unlike most agent communication architectures, KAoS explicitly takes into account not only the individual message, but also the various sequences of messages in which it may occur. Shared knowledge about message sequencing conventions (conversation policies) enables agents to coordinate frequently recurring interactions of a routine nature simply and predictably.

KAoS Architecture

The KAoS architecture currently aims to provide the following:

• A form of agent-oriented programming, based on a foundation of distributed object technology

• Structured conversations between agents, which may preserve their state over time

• An approach for extending the language of inter-agent communication in a principled manner, taking into account the repertoire of illocutionary acts ("verbs") available to agents, the set of conversation policies available to agents, the content of messages, and a means for agents to locate and access desired services

• A framework supporting interoperability with other agent implementations as well as with non-agent programs

• An environment in which to design agents to engage in specialized suites of interactions

agent technologies explained

The strength of the architecture derives from several sources:

- It is built on a foundation of distributed object technology and is optimized to work with component integration architectures such as Open-Doc, ActiveX, and Java and with distributed object services such as those provided by CORBA, DCOM, AND THE INTERNET
- It supports structured conversations that preserve and make use of the context of agent communication at a higher level than single messages; allow differential handling of messages depending on the particular conversation policy and the place in the conversation where the message occurs and permit built-in generic handlers for common negotiation processes such as countering
- It allows the language of inter-agent communication to be extended in a principled manner, permitting verbs and conversation policies to be straightforwardly reused, adapted, or specialized for new situations
- It groups related sets of conversation policies into suites supporting a coherent set services
- It provides facilities for service names (yellow pages), which are advertised to the Matchmaker by agents offering services
- It provides facilities for agent names (white pages), which allow a Domain Manager to uniquely identify an agent as long as it persists
- It is appropriate for a wide variety of domains and implementation approaches and is platform- and language-neutral
- It allows simple agents to be straightforwardly implemented, while providing the requisite hooks to develop more complex ones
- It supports both procedural and declarative semantics

agent technologies explained

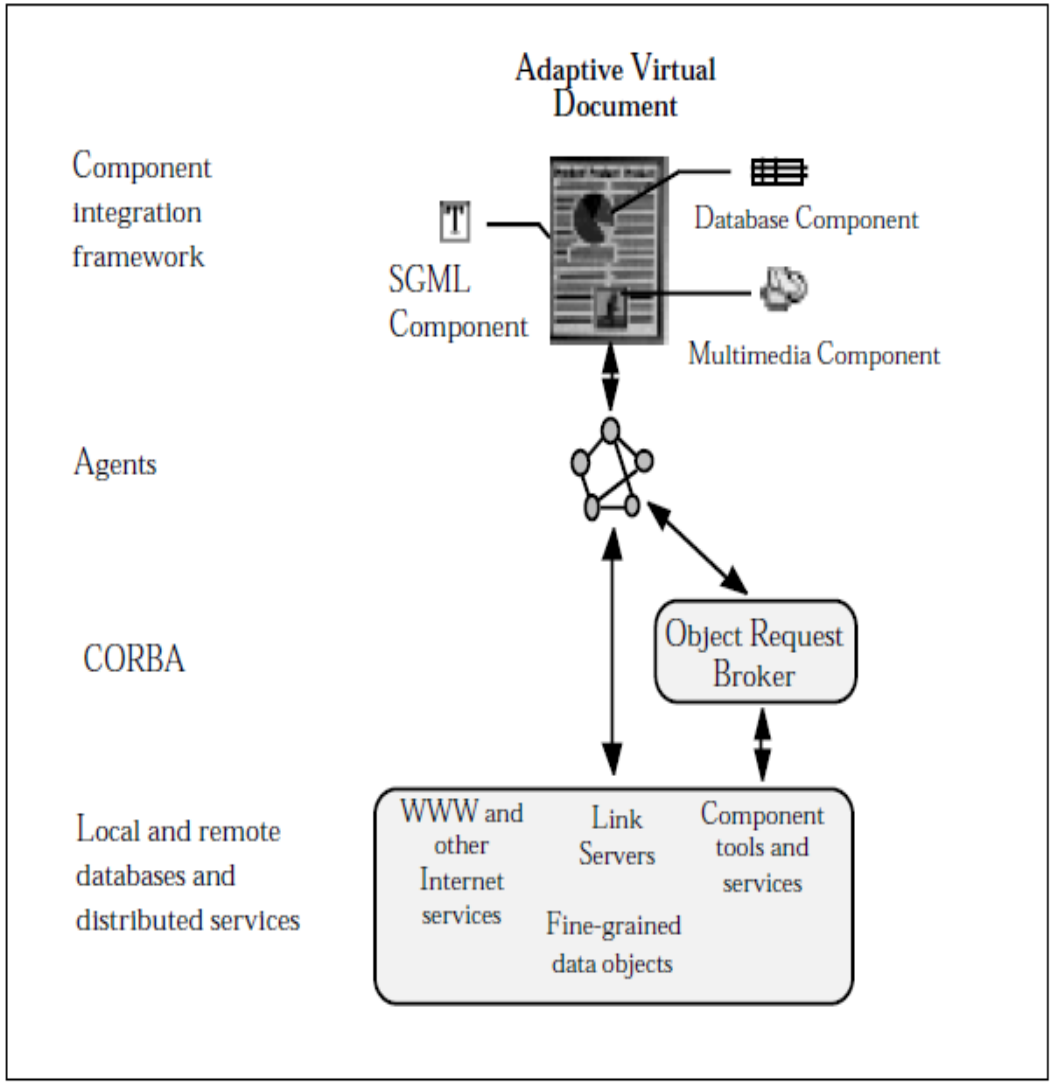


Figure 1. The context in which the KAoS agent architecture is being defined.

- Each agent has **a generic agent** instance, which implements as a minimum the basic infrastructure for agent communication. Specific extensions and capabilities can be added to the basic structure and protocols through standard object-oriented mechanisms.
- **Mediation agents** provide an interface between a KAoS agent environment and external non-agent entities, resources, or agent frameworks.
- **Proxy agents** extend the scope of the agent-to-agent protocol beyond a particular agent domain.
- **The Domain Manager** carries off policies set by a human administrator, such as keeping track of agents that enter and exit the domain. The Matchmaker can access information about the location of the generic agent instance for any agent that has advertised its services.

agent technologies explained

Basic Characteristics of KAoS Agents

- ***Agent-oriented programming*** - an agent can be thought of as an extension of the object-oriented programming approach, where the objects are typically somewhat autonomous and flexibly goal-directed, respond appropriately to some basic set of speech acts (e.g., request,
offer, promise), and ideally act in a way that is consistent with certain desirable conventions of human interaction such as honesty and non-capriciousness. From this perspective, an agent is essentially "an object with an attitude."
- ***Capabilities*** are the services or functions that an agent can provide as defined in specific extensions to the generic agent implementation. The Java implementation of KAoS currently relies on sockets. Because KAoS relies on popular messaging schemes for communication between extensions and the generic agent, agent capabilities can be defined or extended straightforwardly using any combination of standard programming languages, general purpose scripting languages (e.g., AppleScript, Visual Basic, Tcl, Perl, JavaScript) and declarative logic-based programming languages (e.g., KIF, Prolog).
- ***Persistent*** - KAoS agents that are declared as persistent must be able to go into a form of "suspended animation" (called *cryogenic state). Each persistent agent is responsible* for saving the aspects of its structure required allow it to be reactivated when required. The process of saving and restoring structure may also be simple or complex, depending on the situation

agent technologies explained

Agent Dynamics

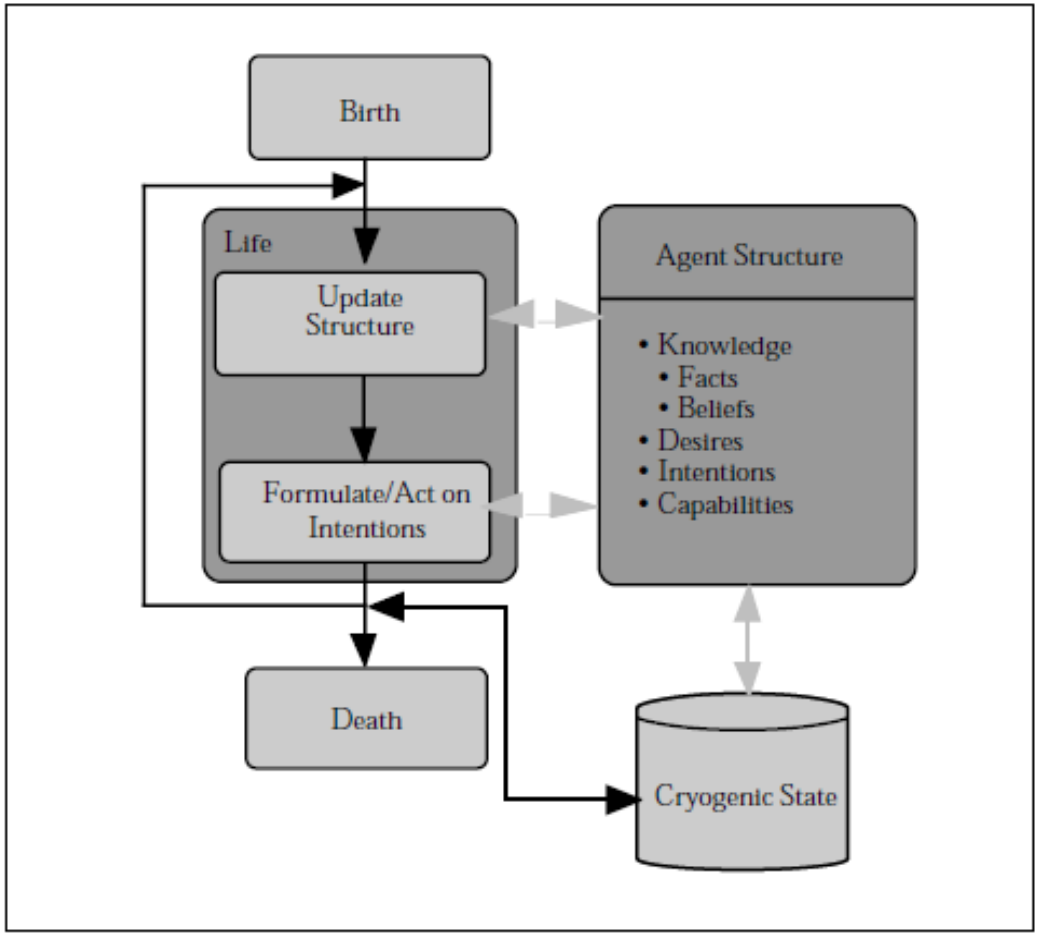


Figure 2. Structure and dynamics of agents (modified from George [1994]). The black arrows represent state transitions, and the gray arrows data flows.

- Figure 2 shows how each agent goes through the equivalent of birth, life, and death.
- At birth, agents instantiated and initialized with some amount of innate structure.
- During their lives, agents go through a continuous cycle of reading, processing, and sending messages. Agents may acquire additional knowledge, desires, and capabilities as they interact with other agents and with their environment. As messages come in, agents update their structure, formulate their intentions, and send new messages in order to act on them.
- In specific applications, agent death may be required to free resources or simply deal with agents that are no longer useful. Agent death poses special problems.

Depending on the application, it may be necessary to include domain specific procedures for dealing with it. These may include notification of other agents, transfer of any pending commitments, or transfer of knowledge.

agent technologies explained

Agents and Objects

- Communication between agents takes place through the use of *messages.*
- *A message consists of a packet of information,* usually sent asynchronously, whose type is represented by a *verb corresponding* to some kind of illocutionary act (e.g., *request, inform)*
- *Messages are exchanged* by agents in the context of *conversations. Each message is part of an extensible* protocol—consisting of both message names and conversation policies—common to the agents participating in the conversation.
- Table 1 enumerates distinctions between communication in classical object-oriented programming and in the agent-oriented architecture.
- The combination of all these features allows agents to analyze, route, and deliver messages properly without necessarily requiring interpretation of content until they reach their final destination.

	Objects	Agents
Basic unit	instance	agent
State-defining parameters	unconstrained	knowledge, desires, intentions, capabilities,...
Process of computation	operations	messages
Message types	defined in classes	defined in suites
Message sequences	implicit	defined in conversations
Social conventions	none	honesty, consistency,...

Table 1. Objects versus agents

agent technologies explained

- Table 2 identifies the characteristics of an operation and compares these to the characteristics of a message.
- By "operation," we mean the invocation of a procedure or a method.
- Though operations may take place in isolation, messages occur only in the context of a conversation.

Operation	Message
Operation name	Verb
Signature	Conversation Parameters
Return Value	(none)

Table 2. Message characteristics.

agent technologies explained

Applications of KAoS agent technology

❑ Early versions of KAoS were used to build demonstrations of agent-oriented programming and simulations of various agent activities. The first prototype implemented a multi-agent version of **a battleship game**, defining specializations of the generic agent class for one or many cooperating ship captains on each team, a game board Matchmaker, an Excel spreadsheet mediation agent, and a referee (Tockey et al. 1995; Atler et al. 1994).

❑ A maintenance performance support prototype demonstrated how mediation agents could **help coordinate the interaction between airline maintenance mechanics and their supervisors** and adapt the presentation of task-related information through a dynamic OpenDoc component interface (Bos et al. 1995). Generic agent capability was specialized to create a supervisor agent, a job administration agent, a user administration agent, and a client mediation agent that handled interaction between OpenDoc "clients" and a KAoS agent domain.

❑ A scheduling environment prototype showed how KAoS could be used to implement **assistants to aid in the process of scheduling meetings and meeting rooms** (Barker et al. 1995). A simulation of interaction with the agent system through electronic mail and agent learning of user preferences was also created. The scheduling environment consisted of a set of scheduling agents, a scenario agent, a mail mediation agent handling interaction between a MAPI mail application and the agent domain, and an OLE journaling mediation agent that communicated with Microsoft Excel.

agent technologies explained

Applications of KAoS agent technology

❑ Early versions of KAoS were used to build demonstrations of agent-oriented programming and simulations of various agent activities. The first prototype implemented a multi-agent version of **a battleship game**, defining specializations of the generic agent class for one or many cooperating ship captains on each team, a game board Matchmaker, an Excel spreadsheet mediation agent, and a referee (Tockey et al. 1995; Atler et al. 1994).

❑ A maintenance performance support prototype demonstrated how mediation agents could **help coordinate the interaction between airline maintenance mechanics and their supervisors** and adapt the presentation of task-related information through a dynamic OpenDoc component interface (Bos et al. 1995). Generic agent capability was specialized to create a supervisor agent, a job administration agent, a user administration agent, and a client mediation agent that handled interaction between OpenDoc "clients" and a KAoS agent domain.

❑ A scheduling environment prototype showed how KAoS could be used to implement **assistants to aid in the process of scheduling meetings and meeting rooms** (Barker et al. 1995). A simulation of interaction with the agent system through electronic mail and agent learning of user preferences was also created. The scheduling environment consisted of a set of scheduling agents, a scenario agent, a mail mediation agent handling interaction between a MAPI mail application and the agent domain, and an OLE journaling mediation agent that communicated with Microsoft Excel.

FUTURE DIRECTIONS

1) allowing mobile users of small computing devices to interact with a KAoS agent domain residing on a remote machine
2) integrating the KAoS architecture with mobile agent approaches that permit the physical migration and secure, managed execution of

agent programs on "guest" hosts not belonging to the sender of the agent.

CONCLUSIONS

The KAoS architecture will succeed to the extent that it allows agents to carry out useful work while remaining simple to implement. Although it is still far from complete, our experience with the current KAoS architecture has shown it to be a powerful and flexible basis for diverse types of agent-oriented systems.

Thank you