ISSN 1465-3842

AGENTLINK NEWS 5

Europe's ESPRIT-funded Network of Excellence for agent-based computing

May 2000 www.agentlink.org

Features

Project Reports

Conference Reports

AgentLink/SIG Reports

Previous issues of AgentLink News have included a number of articles describing tools for the development of agent-based software, e.g., April, Grasshopper, Jack, Mozart, and Zeus. In this issue we continue this tradition with two feature articles about JAM and LARS, supplemented by a short description of Grasshopper II. JAM is based on the original BDI (belief, desire, intention) framework but is influenced by pragmatic considerations made in implementations of the Procedural Reasoning System (PRS). LARS, developed by Living Systems AG of Germany, is an agent platform aimed at business applications and is focusing on interoperability, scalability, and security.

This issue also contains two project reports that describe multi-agent system solutions for radar data analysis in large repositories and process management respectively. In addition, we offer a position statement on the (future) role of agents in e-commerce, and, as always, a number of interesting workshop reports.

As you may already know, the proposal for the continuation of AgentLink (AgentLink II) has been approved by the European Commission, leading to a further Euro 803K funding over three years. More information about AgentLink II can be found inside. The first main event in the life of AgentLink II will be a follow-up to last year's successful summer school on agent systems, to be held in Saarbrucken, Germany, in August this year. You will find detailed information about the summer school inside.

Paul Davidsson, Editor

JAM: A BDI-Theoretic Mobile Agent Architecture

Marcus J. Huber
Intelligent Reasoning Systems, California, USA
marcush@home.com

JAM is a hybrid intelligent agent architecture that provides rich and extensive plan and procedural representations, metalevel and utility-based reasoning over multiple simultaneous goals, and goal-driven and event-driven behavior that are a combination of our own ideas and those of others. JAM draws upon the theories and ideas of the Procedural Reasoning System (PRS), Structured Circuit Semantics (SCS), and Act plan interlingua. JAM also draws upon the implementation pragmatics of the University of Michigan's and SRI International's implementation of PRS (UMPRS and PRS-CL, respectively). This article briefly describes JAM's major features and characteristics.

We developed the JAM intelligent agent architecture as the next step in the evolution of pragmatic BDI-based agent architectures. Our approach to agent architectural theories and design has been to start with a BDI-theoretic core followed by incremental improvements based upon incorporation or modification based on the research of others and our own ongoing research and experience. JAM combines what we believe to be the best aspects of several leading-edge intelligent agent frameworks, including the original BDI (belief, desire, intention) theories and specification of the Procedural Reasoning System (PRS) of Georgeff, Lansky, Rao, and others [2,5], the Structured Circuit Semantics (SCS) representation of Lee and Durfee [7], and the Act plan interlingua [9,12] of Myers, Wilkins and others. In addition, JAM draws upon pragmatics garnered from the PRS implementations of the University of Michigan (called UMPRS) [7]) and SRI International (called PRS-CL) [10].

Starting with a BDI-theoretic "kernel" allows us to reap the benefits of a large body of research on the theory and implementation of, in particular, the Procedural Reasoning System (PRS). Explicit modeling of the concepts of *beliefs*, *goals* (desires), and *intentions* within an agent architecture provides a number of advantages, including facilitating use of declarative representations for each of these concepts. The use of explicit declarative representations in turn facilitates the automated generation, manipulation, and communication of these representations. Other advantages of starting with a PRS-based BDI architecture include a sound model of goal-driven and data-driven task achievement, metalevel reasoning, and procedurally specified behavior. And although capabilities are not a central attribute of BDI theories, implemented architectures typically include explicit, declarative modeling of the capabilities of the agent in the form of plans and primitive actions.

Our research began with the development of UMPRS starting in 1992 to provide us with a well-founded means of encoding behavior for mobile robots [7] for performing mobile robot research and since has been used for a number of additional application areas [1,4,6,11]. UMPRS implements a majority of the representational and behavioral concepts found in the original PRS specification and was extended to provide a superior, more structured procedural representation. UMPRS lacks a strong architectural conceptualization of goals (not suffered by PRS or PRS-CL) however, and lacks some significant procedural constructs (e.g., parallel execution) and goal types (e.g., homeostatic goals). JAM does not suffer from these shortcomings, and provides strong goal-achievement syntax and semantics with support for homeostatic goals and a much richer, more expressive set of procedural constructs.

The stronger, formalized goal implementation of JAM is motivated by work in progress on extensions to support generative planning capabilities within the JAM architecture and originate most directly from the original PRS specification and the large body of classic AI planning research. Procedural extensions were garnered from the Structured Circuit Semantics (SCS) representation [8] and the Act plan interlingua [9,12]. Functional improvements such as agent mobility are founded in the research from the Dartmouth and IBM mobile agent research efforts [3].

JAM has been used in projects by a number of institutions and universities around the world. These include Johns Hopkins University's Applied Physics Laboratory, ORINCON Corporation, DFKI, Honeywell Corporation, MIT's Media Lab, and Seoul University.

JAM architectural overview

Each JAM agent is composed of five primary components: a *world model*, a *plan library*, an *interpreter*, an *intention structure*, and an *observer*. We illustrate this in Figure 1. Other than the observer, the rest of the components are fairly standard PRS components. The world model is a database that represents the beliefs of the agent. The plan library is a collection of plans that the agent can use to achieve its goals. The interpreter is the agent's "brains" that reason about what the agent should do and when and how to do it. The intention structure is an internal model of the agent's current goals and keeps track of the commitment to, and progress on, accomplishment of those goals. The observer is a user-specified lightweight declarative procedure that the agent interleaves between plan steps in order to perform simple periodic functionality (e.g., to transfer incoming messages to JAM's world model).

The JAM execution semantics and behavior is a combination of that of UMPRS and SCS. Changes to the world model or posting of new goals triggers

reasoning to search for plans that might be applied to the current situation. The JAM interpreter selects one plan from this list of applicable plans based on either metalevel reasoning or maximum utility, *intends* it (i.e., commits itself to execution of the instantiated plan), and executes the first intention found with the highest utility (i.e., as if there was an SCS DO_BEST over all top-level goals). The remainder of this section discusses each of the major components of JAM in as much detail as space permits.[1]

Interpreter

The JAM interpreter is responsible for selecting and executing plans based upon the agent's current intentions, plans, goals, and beliefs. Associated with the interpreter is the *intention structure*, a run-time stack (stacked based upon subgoaling) of goals with and without instantiated plans. A JAM agent may have a possibly large number of alternative plans for accomplishing any single goal and the JAM interpreter reasons about all of the alternatives combinations of plans, goals, and variable bindings (based on goal arguments and beliefs) before selecting the best alternative given the particular situation.

The agent checks all the plans that can be applied to a goal to make sure they are relevant to the current situation. A utility value is determined for each instantiated plan in the APL and, if no metalevel plans are available to select between the APL elements, the JAM interpreter selects the highest utility instantiated plan (called an *intention*) and *intends* it to the goal. Note that neither the original PRS specification nor prior PRS-based implementations (such as PRC-CL) supports utility-based reasoning. If the goal with the new intention has the highest utility among all goals with intentions, then the new goal's plan is executed. Otherwise, a previous intention still has a higher utility and the interpreter executes that intention's plan.
If generation of an APL results in one or more entries, the agent may enter *metalevel* reasoning. That is, it reasons about how to decide which of the APL elements to intend to its intention structure. The agent may have multiple applicable metalevel plans of performing this decision-making, so that metalevel reasoning about its metalevel reasoning may ensue. Metalevel reasoning ends when the interpreter no longer generates a non-null APL, indicating that the agent has no higher metalevel means for deciding between alternatives. The JAM distribution comes with a number of implemented primitive actions to facilitate metalevel reasoning.

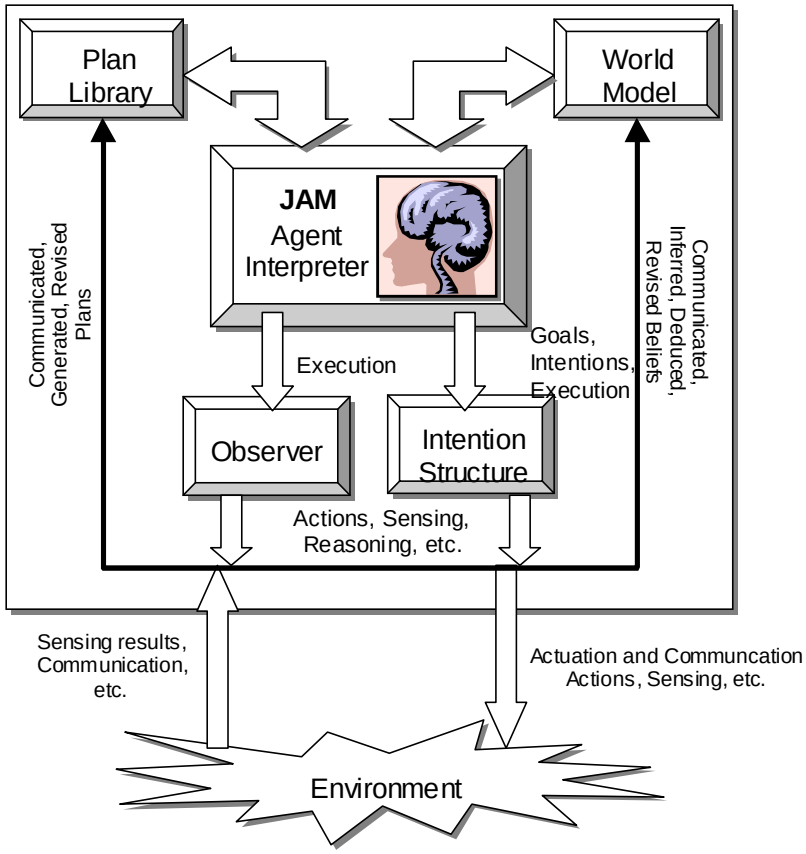

Figure 1: The JAM Intelligent Agent Architecture

Goals

A JAM agent's top-down behavior is motivated by specifying top-level goals. A goal is one of ACHIEVE, PERFORM, or MAINTAIN. A goal itself is a relation, specified by a relation name and an ordered set of arguments. A JAM agent will dynamically switch between alternative goals as the intention utilities change so that it is always pursuing the highest utility intention (hence, goal). One or more top-level goals are initially given to the agent at agent invocation. This list of goals can be augmented during execution by such means as communication with other agents and generated from internal reasoning.

JAM supports a number of different types of goals - ACHIEVE, PERFORM, and MAINTAIN - each with distinct semantics. An ACHIEVE goal specifies that the agent desires to achieve a goal state and is the goal type typically associated with BDI architectures and generative planning systems. A world model entry indicating that the goal has been achieved is asserted if the plan selected for accomplishment of the ACHIEVE goal completes successfully. Subsequent attempts to ACHIEVE the same goal relation will not result in the agent intending action as long as this world model relation exists in the agent's "head".

A PERFORM goal specifies the agent desires to execute some behavior. This semantics is an extension not found in typical BDI architectures and indicates that the agent is not interested in achieving a goal, per se, but merely to *do something*. Subsequent attempts to achieve the same PERFORM goal will result in the agent identifying and intending application plans to fulfill the goal.
A MAINTAIN goal indicates that the specified goal must be re-attained if it ever becomes unsatisfied. A MAINTAIN goal is similar to an ACHIEVE goal except that a MAINTAIN goal is never removed from the agent's goal list automatically (i.e., it is a *homeostatic* goal). A MAINTAIN goal can be removed from the agent's intention

structure only by the agent explicitly removing it.

Plans

A JAM plan defines a procedural specification for achieving a goal state, reacting to an event, or performing behavior. JAM agents are therefore capable of both goal-driven and data-driven behavior. One or more plans are initially given to the agent at agent invocation. This list of plans can be augmented during execution through communication with other agents, generated from internal reasoning on the part of the agent, or by many other means. A plan's applicability is limited to either a particular goal or a data-driven conclusion. Each plan may be further constrained to a particular precondition and context, conditions that must hold before starting execution of the plan and conditions that must hold both before and during execution of the plan, respectively. This semantic differentiation of runtime and pre-runtime conditions provides more flexibility to an agent programmer than does the context semantics found in the PRS or Act specifications. The procedure to use to accomplish the goal is given in the plan's procedural body, which can contain simple actions (e.g., execute a user-defined primitive function) and complex structured constructs (e.g., iteration, conditional branching). Each plan may include an explicitly or implicitly defined utility calculation, which is used to influence selection of certain intentions over others through JAM's default utility-based scheme. A plan's effects field is a procedure that the JAM interpreter executes when the plan completes successfully. A procedural specification of what the agent should do when a plan fails can be represented in a plan's optional failure section. The optional plan attributes field provides a place for a programmer to put information concerning plan characteristics that the agent can reason about during plan execution and metalevel reasoning.

Agent programmers can augment the functionality provided with JAM by defining primitive functions in native Java code and several access methods to the Java code are provided by the JAM architecture. It is through this augmentation of primitive functions that provide JAM with application-specific and social abilities (e.g., database interfacing and interagent communication and collaboration). There are a number of predefined primitive actions included with the JAM agent distribution, including those providing debugging support and agent mobility (which we describe in more detail below).

World model

The JAM World Model holds the facts that represent the current state of the world as it is known by the agent. Information that might be kept there includes state variables, sensory information, conclusions from deduction or inferencing performed within plans, and modeling information about other agents. Each world model entry is a simple proposition, a relation with an ordered list of arguments.

Observer

The *observer* is an optional declarative procedure that the JAM interpreter executes between each action in a plan. The observer procedure itself is basically a plan with only the plan's procedural body and none of the other plan components. The observer represents an architectural hook which an agent programmer can use to implement asynchronous capabilities. We named this procedure "observer" because of its typical use in watching for asynchronous events. Because the observer procedure is executed very frequently, it should not be computationally intensive. Note that all of the functionality and behavior embedded into an observer procedure can be implemented in JAM's normal BDI paradigm (using goals and plans) to take advantage of the architecture's powerful reasoning capabilities. We have found through experience that some simple functions, such as moving messages from low-level buffers to the agent's World Model, are better accomplished using the observer than going through the complexities of JAM's full reasoning scheme.

JAM agent checkpointing and mobility

JAM agents facilitate building applications requiring mobility through the use of *checkpointing* capabilities. That is, we have implemented functionality for capturing the runtime state of a JAM agent in the middle of execution and functionality for subsequently restoring that captured state to its execution state. One use of this functionality is for periodically saving the agent's state so that it can be restored in case the agent fails unexpectedly. This facilitates building robust applications that can restart and recover from otherwise catastrophic termination. Another use of the checkpointing functionality is to implement agent mobility, where the agent creates a checkpoint and restores it to execution on a different computer. A third possible use of this functionality is to clone an agent by creating a checkpoint and restoring it to an execution state without terminating the original agent. In all cases, a simple Java class is provided with JAM that performs the basic restoration function. Extension of this restoration class to provide application-specific mobility policies and similar functionality can be made as needed.

We have simplified agent mobility by implementing an *agentGo* primitive function. This function allows an agent programmer to simply specify a target computer and port and, when the plan containing the function is executed, the agent will transfer to the other machine and terminate itself on the initial computer. On the destination machine, the JAM agent will resume execution, guided by its pre-existing goals and plans, from where it was suspended on the initial machine. Moving between computers therefore becomes transparent in the sense that such activity is not handled differently than any other activity. The example plan shown in Section 2.3 illustrates a plan using the agentGo primitive.

Conclusions and future work

We believe JAM represents the current leading edge in pragmatic BDI-theoretic intelligent agent architectures. JAM provides rich, expressive procedural representations, a wide range of useful goal semantics, metalevel reasoning, support for complex utility-theoretic behavior, and agent mobility support while remaining true to its BDI underpinnings.

JAM does not represent a complete architecture yet, however, in that many architecturally integrated capabilities such as plan generation and learning do not yet exist. Towards the end of a JAM architecture that includes these capabilities, we are currently in the middle of adding generative planning functionality to the JAM interpreter, so that when JAM reaches an impasse (in Soar terminology), it can generate a plan from first principles (using a novel hybrid HTN and partial order planning algorithm) rather

than relying solely upon the library of pre-programmed plans as most BDI architectures (e.g., PRS-CL and UMPRS). We have extended the JAM plan representation to include declarative representations for individual primitive actions and are implementing partial order planning algorithms based upon the new representations. "Social" abilities in the form of conversation management and FIPA-compliant language and protocol support have been added to JAM as an application-specific extension, but we have not yet decided upon whether such capabilities will become an integral part of JAM at any point or if such functionality will be provided as a supplemental package.

References

[1] E. H. Durfee, M. J. Huber, M. Kurnow, and J. Lee. TAIPE: Tactical Assistants for Interaction Planning and Execution. In *Proceedings of the First International Conference on Autonomous Agents*, 443-450, Marina del Rey, CA, 1997.

[2] M. Georgeff and A. L. Lansky. Reactive Reasoning and Planning. In *Proceedings of the Sixth National Conference on Artificial Intelligence*, 677-682, Seattle, Washington, 1987.

[3] R. S. Gray, D. Kotz, G. Cybenko, and D. Rus. Agent Tcl. In W. Cockayne and M. Zyda editors, *Mobile Agents*, Manning Publishing, 1997.

[4] M. J. Huber and T. Hadley. Multiple Roles, Multiple Teams, Dynamic Environment: Autonomous Netrek Agents. In *Proceedings of the First International Conference on Autonomous Agents*, pages 332-339, Marina del Rey, CA, 1997.

[5] F. Ingrand, M. Georgeff, and A. Rao. An Architecture for Real-Time Reasoning and System Control. *IEEE Expert*, 7(6):34-44, 1992.

[6] P. G. Kenny, E. H. Durfee, and K. C. Kluge. Mission Planning and Coordinated Execution for Unmanned Vehicles. In *Proceedings of the Sixth Computer Generated Forces and Behavioral Representation Conference*, 329-335, 1996.

[7] J. Lee, M. J. Huber, E. H. Durfee, and P. G. Kenny. UM-PRS: An Implementation of the Procedural Reasoning System for Multirobot Applications. In *Conference on Intelligent Robotics in Field, Factory, Service, and Space (CIRFFSS'94),* 842-849, Houston, Texas, 1994.

[8] J. Lee and E. H. Durfee. Structured Circuit Semantics for Reactive Plan Execution Systems. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, 1232-1237, 1994.

[9] K. L. Myers and D. E. Wilkins. The Act Formalism, Version 2.2. SRI International Artificial Intelligence Center Technical Report, Menlo Park, CA, 1997.

[10] K. L. Myers, User Guide for the Procedural Reasoning System. *SRI International AI Center Technical Report*. SRI International, Menlo Park, CA, 1997.

[11] J. M. Vidal and E. H. Durfee. Task Planning Agents in the UMDL. In *Proceedings of the 1995 CIKM Intelligent Information Agents Workshop*, 1995.

[12] D. E. Wilkins and K. L. Myers. A Common Knowledge Representation for Plan Generation and Reactive Execution. In *Journal of Logic and Computation*, vol. 5, number 6, 731-761, 1995.

1 Complete details on JAM can be found at <http://members.home.net/marcush/IRS/Jam/jam-man.doc>.

Article

living agents runtime system (LARS) – the Agent Platform for Business Applications

Norbert Nopper
living systems AG, Donaueschingen, Germany
Norbert.Nopper@living-systems.de

LARS 2.5 is an agent platform developed by living systems AG. The development of this Java [1] based agent server and the *living agents* – the agents running on this platform - began in 1997. The system has been successfully displayed and utilised in research projects such as AMTRAS [2] for the German Stock Exchange (OTC trading) or ENTRAS [3] (B2B energy trading) and business projects for eBay.de, AsiaOne and NetBid, to mention just a few. The purpose of this paper is to introduce the *LARS 2.5* platform, and to take a closer look at the *living agents*. Finally, there is a brief introduction to the *living agents* explaining the benefits of *LARS 2.5* usage in business applications.

The *living agents runtime system* was designed with the goal of implementing an agent platform with the additional agents that would be:

- operation system independent
- highly scalable
- standardised in communication
- secure

The points listed above, and the corresponding techniques are explained in the first section of this paper. The second

section discusses the platform's usage and the agents' functionality.

Techniques

The following subsections explain the different techniques that fulfil the design goals.

Operation system independent The operating area for *LARS* is the Internet, or an Intranet. Unfortunately, we have a heterogeneous network of different computers and operating systems. The solution for having a platform independent system, and to keep up communication between these platforms (described later in this section), is Java.

Figure 1 displays the different architecture layers of *LARS*. *LARS* provides the environment for the *living agents* to run (business and system agents). This includes starting, reloading and stopping, priority setting and monitoring, access to system resources of agents, to mention just a few. It also provides security and communication functionality which will be described later.

Because the living agents are written entirely in Java, it is possible to run them on any machine which has an implementation of a Java Virtual Machine 1.2.x or above. Our system has been tested on Windows, Linux, Digital Unix and Sun Solaris. living systems uses Sun Solaris and/or Linux.

Highly scalable To fulfil the requirement of having a fast and stable environment for a business application, the system has to be highly scalable. There are different levels for scaling a *LARS* cluster (see Figure 2).

Firstly, for each platform request, there can be more than one agent on one

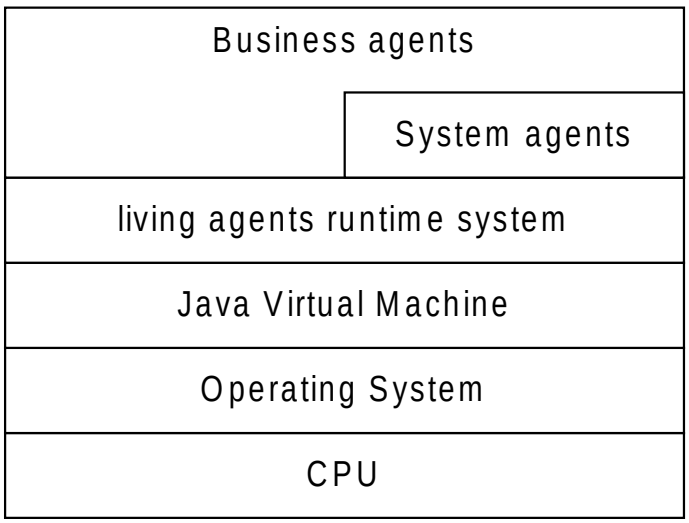

Figure 1: Architecture Layers

LARS platform, which responds to the request. Each agent is running as a separate thread, so the first level of scaling is running multiple agents concurrently on one *LARS* platform. The second scalability level runs multiple *LARS* platforms on a multiprocessor computer. Therefore, each platform runs independently on one processor. Because each *LARS* platform can be connected with one another (described later in this section), it is possible to add one *LARS* platform dynamically, and to transfer one agent from one *LARS* to the other *LARS* platform during the runtime. We use the mobility of agents to achieve load balancing.

The next level of scalability runs different *LARS* platforms on different machines with different operating systems. Here, the same advantages such as the possibility of running multiple *LARS* platforms on one computer are present. Additionally, one can build a cluster with different operating systems and machines. The benefit is the small step of additional machine power in the whole cluster. To achieve such scalability, you need a clean implementation of communication, which is explained in the following section.

Standardised communication The communication of the *living agents* is a very important topic, and has made an evolution from KQML [4] over ACL [5] to XML [6]. One of the advantages of using XML is that the agent communication language will easily build interfaces with 3rd party products.

All communications between the *living agents* use XML. Each *living agent* has a strictly defined protocol, in which messages are understood and through which replies come back. The programmer must fulfil the communication interfaces of the other agent. He/she is not responsible for synchronisation or other agent system resources, or even on which platform the agent runs.

To have a fast and at the same time open communication interface, the transportation of XML can be done using different techniques.

If the agents are on one platform, the XML message is passed by reference, quickest. If there is a communication over a slow connection, or the other side does not understand Java XML objects, the XML is passed as a String over Sockets. Otherwise, the communication between the platforms is with RMI or Corba [7]. To prevent communication over the Internet from attacks, security is a very important issue, especially in business.

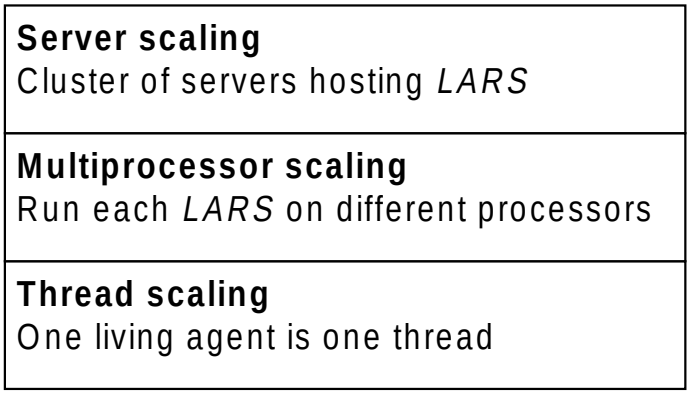

Figure 2: Three Levels of Scaling

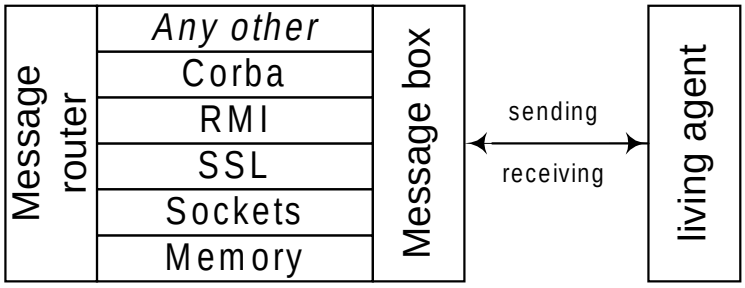

Figure 3: Communication Interfaces

Security The most common way of bringing information to a user is currently by using HTML. To make this information secure, SSL is used and the business/ application logic is located behind a firewall. But future clients won't just be using HTML; B2B affords many possibilities, and it has to be possible to use push technology.

A B2B trading platform can work on a virtual private network, and can be secured with firewalls and IP-denied by the servers. But additional securing mechanisms are needed which *LARS* provides.

As described in Figure 3, one of the communication layers is SSL. Every message that goes over this communication channel is encrypted. This means that if a message leaves a *LARS* platform, the whole message is encrypted. An additional feature of the security layer of *LARS* is the possibility to encrypt specific fields and/or the sub tree in an XML message. This can be used in applications where only specific fields are confidential, and speed is more important. You can still encrypt the whole message, and here each customer can use its own encryption algorithm. Another important feature is the functionality of signing a message. With the public/ private key method, it is possible to check the originator of a message. This functionality is very important for any transactions over the Internet. Finally, there is a PIN/TAN system for transaction systems, to exclude any invalid messages or transactions.

The second section of this paper will demonstrate how our *living agents* can be used in business applications.

Agents in the business

living agents can be used for different purposes in the business. One implementation is to use the *living agents* for auctions. The following chapter focuses on agents and auctions in business. living systems AG provides an extension module for its agents, the *living auctions* functionality. This means the agents can handle auction models like the English Auction, Dutch Auction or the Vickrey Auction. Basically there are two different types of agents: bidder and auctioneer. It is also possible to have both functionalities in one agent.

C2C In a C2C market place, an auctioneer agent is responsible for the users auctions: bidding synchronization and auction ending. Besides placing a bid on an auction, the user can also place a proxy bid with an agent. The agent bids with the lowest price and the agent automatically increases the bid if another user has placed a higher bid. To handle millions and millions of users, auctions and bids in C2C, one agent is responsible for ending and synchronizing several auctions from different users. The proxy bids from all users is also shared over several agents. There is no reason to represent one auction by one auctioneer and a proxy bid by one agent. The parameters are the end of the auction and the bid limit of each agent or user. Additionally, the duration of each auction is at least three days.

However the functionality mentioned above is only a small percentage of the whole *living auctions* system. The benefit of the *living auctions* functionality can be used in B2B market places.

B2B The *living agents* and the *living auction* module make B2B market places possible. The same mechanism mentioned above can be used in B2B of course. The main benefit of trading with agents is the possibility of having a personalised agent for each trader or company.

Personalisation begins with finding and matching necessary counterparts. The parameters do not have to match exactly. Each trader can configure his/her own agent, which focuses on particular topics. This "soft matching" reduces time, because counterparts are found quicker by the agents. The agents' benefit is the speedier filtering and delivering of many counterparts.

At this point, a trader can begin discussions about price, which can take a long time and cannot be accomplished with many users participating at the same time. However the *living agent* can. If an agent has located counterparts, not only is it able to trade with one agent after another, the agent can trade with all agents at the same time. Additionally, if the traders desire it, each state can influence the other trading state and the complete strategy of the agent.

One strategy would be to give the agent the functionality of increasing its next bid depending on the last bid of the counterpart. Triggers in changing a strategy would be a better option in another trade or if a new trader appears.

The main benefit in using the *living agents* and *living auctions* system is that trading can take place now, which could not before, (e.g. OTC) and that this can happen in milliseconds.

Summary

The *LARS* 2.5 system provides a robust, fast and secure platform, needed for any serious application. Each additional *living agent* increases the powerful library of *living agents*. Every agent is able to communicate with one other, if they implement the defined communication language. Adding or replacing an agent during run time is possible. The communication is open to third party products through XML.

Because every *living agent* is a specialist in its domain, it is possible to fulfil requirements in fast changing markets. The system is open for the future because each new *living agent*, which is a specialist in its domain, can be added immediately and can rely on the existing system of today.

About *living systems*

With a yearly growth rate of more than 100%, living systems AG [8] has become

one of the strongest and fastest growing internet technology corporations. Since the summer of 1999, the company has founded offices in London, Singapore, Sao Paulo and Boston.

The company's application focus lies increasingly in vertical and horizontal B2B marketplace developments where dynamic pricing mechanisms are being applied. The most recent example of a living systems' application prototype is the ENTRAS trading system developed for B2B energy trading. The Baden-Württemberg e.V. Electricity association (VdeW) awarded ENTRAS in January 2000, the most innovative application in energy trading. In February, living systems AG received the "Most Innovative Company" award out of 300 competitors in Baden-Württemberg, for extraordinary company performance.

The company's reference list is diverse with such successes as BMW, SairGroup/ Atraxis and Wrigley, and Internet start-ups like www.ebay.de, www.dooyoo.de, www.yoolia.com, www.tradenetone.com, and NetBid Industrie Auktionen AG, www.netbid.de.

References

[1] Java. The Source for Java Technology <http://www.javasoft.com>

[2] AMTRAS. Informationssysteme in der Finazwirtschaft Springer, 1998

[3] ENTRAS. Elektronischer Stromhandel mit intelligenten Software-Agenten Diplomarbeit an der FH Furtwangen, Fachbereich Wirtschafts-Informatik, 1999

[4] The DARPA Knowledge Sharing Initiative (1993). Specification of the KQML <http://www.cs.umbc.edu/kqml/kqmlspec.ps>

[5] FIPA. Agent communication language <http://www.fipa.org/spec/f8a22.zip>

[6] XML. The XML industry portal <http://www.xml.org>

[7] Corba. The OMG's site for Corba <http://www.corba.org>

[8] living systems AG <http://www.living-systems.com>

Towards an Online Distribution Structure?

Kees Jonkheer
NetlinQ Groep N.V., The Netherlands
kees@netlinq.nl

For many years intelligent agents have been seen as holding much promise for e-business. Applications of web intelligence are software programs that execute functions on behalf of a user on the internet, such as searching, comparing, learning, negotiating and collaborating. Intelligent agents combine at least a few of these functions. In the Netherlands agents are receiving increasing attention, good examples being the products of agent builders Tryllian and Smarthaven. Tryllian recently launched the Gossip-agent and has been successful with it in the United States, where huge interest exists for a commercial application of this concept. Tryllian's ambitions are high: their aim is to set a world standard for agent platforms. The fact that Stichting NLnet and the Vrije Universiteit of Amsterdam recently obtained a ten-year budget for a research project aimed at the application of intelligent agents to privacy and security issues is indicative of the interest and the importance that is currently attributed to agent technology. The emergence of e-commerce on a massive scale, the mobile internet and the ever growing need for businesses to improve customer contact mean a definitive breakthrough for web intelligence applications and agent technology in particular. Moreover, renewed interest in agents corresponds with the quest to find a fitting interface between the user and the WWW. Within 5 years it might be hopelessly outdated to surf the web - will every business have its own web site, visited by endlessly searching and clicking customers? In e-business there appears to be a huge gap in the value added of online fulfilment. The aim should be to minimise the so-called sacrifice gap; the difference between what customers want and what they eventually get. Agents will offer a solution to this problem, and will become a discriminating factor in the evolution of an online distribution structure.

Large-scale e-commerce

Large-scale e-commerce stimulates the need for and possibilities of applications of web intelligence and more specifically, of agent technology. This can be illustrated by the following example. Because of massive employment in e-commerce, the demography of online buyers in the United States has changed dramatically over the last two years. Initially it was predominantly higher educated males aged between 20 and 35 who bought on the net; now online buyers are found in every segment of society. Whilst the first group were probably buying online out of convenience and curiosity, price is now the driving factor for web consumers. This explains why shopping bots and price comparing sites with bot-functionality have become very popular over the last two years. Software builder Autonomy sees its existence as being largely based on massive Internet use and considers itself 'the Oracle of unstructured data'. Two main forces are push factors for the usability of their products: personalisation of service and the information-explosion in text-form (*WIRED* 8.02, February

2000). Autonomy use so-called 'Concept Agents' that on the basis of context analysis retrieve information from a huge amount of unstructured data. Autonomy's Easdaq stock price has risen a thousand percent since the IPO in July 1998.

Customer contact

Websites have until now been strictly self-service environments. But these self-service environments do not offer the same kind of benefits as the active, skilful and personal care customers receive in the offline world. Companies such as General Motors, Mercedes-Benz, Tower Records and Egghead Software are thus trying to incorporate the qualities of traditional customer care by integrating call centre services on their websites. Call centre personnel can answer questions by using the telephone or Internet chat connections. Although useful for customers, this approach is very expensive, labour intensive, and scalability is a problem. So this calls for different, more automated solutions.

Increasingly, online customers can be offered the right supply at the right moment or through new combinations of services. This interactive approach leads to new forms of value creation. An important form of added value for the customer is the one-to-one approach: the supplier tries to identify the individual customer's needs and offers fully individual propositions. To achieve optimal online individual servicing, some companies are employ additional software for call-centre personnel, or offer guaranteed service for their most important customers. Other applications are customer support by several forms of mass customisation, for example: expert systems, collaborative filtering systems, profile-based information push and question-answer agents. The number of online Customer Relations Management Tools is growing rapidly, with a central focus on the introduction of an interface that should make it easier for customers to execute actions such as navigation on the web site. Examples of companies that offer such products are Peoplesupport, Neuromedia, Chatrep and Artificial Life, who offer many different kinds of customer contact technologies, some of which are approachable in real time (e.g. 'push-to-talk' buttons).

It is obvious that developers of agent technology see an important role for their products in this area. Artificial Life sell Web Guide, one of its web intelligent 'agent-like' products, to e-commerce retailers, financial services, consultancy bureaus and content providers. PriceWaterhouseCoopers use Artificial Life's 'Personal Tutor' for internal knowledge management. Within months Domega, the forthcoming Dutch portal for home improvement on which 200 000 products will be sold, will introduce a digital passport for users. A logical next step will be to connect autonomous action functionality to the digital passport, so incorporating the functionality of intelligent agents.

Mobile internet

In a report published in October 1999, the OVUM research institute predict that intelligent agents will be the primary interface between the WWW and the user in the future. The market for personal assistant services and intelligent agents for mobile phones will be worth up to 26 billion dollars in 2005. With a combination of WAP enabled browsers and personal agents a new personal interface will evolve that will form a personal mobile portal on the Internet. The report even states that agents will eventually be the sole interface between users and the WWW. Instead of customers entering the Internet to buy products, agents will login on the Internet to order and pay for the products.

The combination of intelligent agents and mobile devices is currently put into practise by AlphaServe.com, a US stock listed company who have developed intelligent agents for the Palm VII. Making use of their Network Query Language, their intelligent agents can execute several tasks such as filling in forms, gathering information and making transactions. Autonomy have also made the connection with mobile devices with their i-WAP technology.

Customers offline more and more important

Customers are more and more committed in the production process. A relatively new example of this is Demand Chain Management, which is considered as a form of Supply Chain Management. But this time the producer will have as close as possible a relationship with the customer in terms of attention and client information, that it fully focuses on product development of the individually customised product. Because of this, functions in the physical distribution chain move upwards. The actual production will be done by the former main supplier. The function of the main supplier will be executed by former supplier and jobbers, etc.

In such a concept there has to be a real time customer approach (for example a real time chat partner, such as an online product advisor), that can test, incorporate preferences, etc. in order to offer the customer an individual customised product. The customisation of a product becomes increasingly important in business-to-consumer relations, but is probably even more important in business-to-business relations. There, the customer is a producer too, who uses the purchased product in its own production. Applicability of the product is thus crucial.

The physical distribution chain recognises the importance of communicative nearness to a customer. In an 'online distribution chain' such a role can be executed very well by intelligent agents. Partly, because it is impossible for companies to maintain optimal customer relations aside of their core businesses and partly because intelligent agents are capable of integral efficient information supply and can execute all kinds of tasks.

Towards an online distribution structure?

Distance and added value are discriminating factors in the physical distribution structure. An example: a farmer produces milk, the milk plant pasteurises and packs the milk and supermarkets present the milk to the customers. To buy milk at the source, it would take the customer too much time and trouble to obtain the product and it will be deprived of added value (pasteurisation and packing). Thus, the distribution chain takes care of delivering the product as quickly and conveniently as possible to the customer.

Similarly, online products also need to be prepared for consumption. Discriminators here are information and added value. Distance (in the physical world) and information (in the virtual world) are comparable entities, then. Distance can be overcome if it really has to be, just as information can be found if it really is necessary. But it can be too inefficient and inconvenient in terms of money and time, which makes it impossible to obtain the needed product or service. We consider an online situation largely based

on information distribution, with a focus on buying decisions, and theorise using a 'rational' customer, but do not take trend-skips in preferences or fun shopping into consideration.

The future

The functionality of intelligent agents depends not only on the advancement of the technical state-of-the-art but also on developments that will increase the functionality of current intelligent agents on the web. Greater functionality will mainly manifest itself in, for example, greater opportunities for comparing and learning, as well as the possibility of making transactions online. A report published in May 1999 called *The New Economics of Transactions: Evolution of Unique e-business Internet Market Spaces* (Deloitte Consulting), predicts a new fundamental transactions platform on the Internet as a result of six emerging factors. Three connected forces are: the increasing role of intelligent agents, the evolving phenomenon of the electronic wallet, by which the difference between money and software will disappear, and the emergence of XML and XSL that replace HTML to make more efficiently distinctions between meta data and display data. This distinction is crucial for the autonomous functionality of intelligent agents, because they need meta data in order to function properly.

What will be the application of intelligent agents in the near future? They will fill up the 'open space' in the possibility of supplying online added value. The main question here is what this interface will look like, and how it bridges the gap between the perceived value and expected value of the customer. It is clear that such an interface should be personal, accurate and allow the exhaustive building up an exclusive tradition of trust. That role will be filled by intelligent agents in, for example, the function of a personal agent as a personal portfolio manager. There will also be agents that execute tasks on specific skills such as finance or the purchase of consumer goods. A-Life, for example, recently produced a Smart Bot for the insurance industry, to which a Knowledge Base is connected with branch-specific data or knowledge.

In this way, intelligent agents will play a decisive role in the evolution of an online distribution structure. The chains in the physical distribution structure of producer, whole sale, supermarket and even restaurant, are analogous to the chains in the online distribution structure, such as the company site, portals and intelligent agents. In this structure agents are the pervasive interface between the user and the Internet. Such agents are personal alter egos or representations of companies. As agents focus on certain specialisations, they perform a more independent role. In the online distribution structure it is quite possible that agents might evolve into brands in their own right, by creating individual "identities" (e.g. "agent X gets quality for a bit of a high price").

In short, it is very likely that as a result of massive e-commerce, the crucial role of customer relations and the mobile internet it will just be a matter of time as to when intelligent agents will play a huge, if not a pervasive role in e-business. Indeed, the first commercial applications are entering the market right now.

In January 2000, NetlinQ initiated a Special Interest Group in the Netherlands and Belgium on Web Intelligence. Participants are the companies Artificial Life/MBE, Autonomy, Bolesian, Medialab, NetlinQ, Q-Go, Smarthaven and Tryllian.

Project Report

A Multi-Agent System for Analysing Synthetic Aperture Radar Atlas (SARA) Data

Omer F. Rana, Yanyang Yang, Christos Georgesoupolou
University of Wales, Cardiff, UK (o.f.rana@cs.cf.ac.uk)
David W. Walker
Oak Ridge National Laboratory, Tennessee, USA (walker@msr.epm.ornl.gov)
Roy Williams
California Institute of Technology, USA (roy@caltech.edu)

We describe the use of a multi-agent system for analysing a large data repository, containing both structured and flat-file based data, for the SARA archive. The multi-agent system comprises both intelligent and mobile agents, and a prototype system, developed using the Voyager libraries from ObjectSpace [2], is briefly described.

The Synthetic Aperture Radar Atlas (SARA) is a digital library of multi-spectral remote sensing imagery of the Earth, 40 TB in total, acquired by the space shuttle in 1994/95. The data is partially replicated on disks and tape robots at Caltech, the San Diego SuperComputer Center (SDSC), and at the University of Leece in Italy. The data is maintained in different kinds of file systems, such as Sun NFS, IBM/Livermore HPSS, and delivered using web front ends. The web interfaces act as an integration tool for combining different server implementations. The process works as follows: a user selects a particular part of the Earth, choosing an area where data is available. A URL is generated whose content is the multi-channel data set to be analysed by the active system. The generated URL corresponds to the region of interest as an SAR image. The processed multi-spectral data may be further processed

by choosing a mapping from the frequency/polarisation channels to be red, green and blue components of the final image. This mapping may be optimised to highlight aspects such as ground ecology or snow/ice conditions, for instance. A use-server communicates with a meta-data server, which contain metadata descriptions such as the position of the image on the surface of the Earth, and the use-server opens an HTTP connection to a data-server. The data-server retrieves the requested data from a mass storage system, and deliver the image as JPEG files. Additional compute-servers are provided for performing activities such as image processing and land classification.

The data from a particular request can vary in size from large data objects a few megabytes in size, to small data sets generated by more complex requests. Hence, a request could be a file name and a simple filter to be applied to that file, or an SQL query to a database, whose output is a list of files which satisfy the query. Metadata is available from web-enabled database systems, converting longitude and latitude to dataset ID's, converting these ID's to filenames, and so on. Data and metadata servers are replicated to provide fault tolerance, so that a given request can be diverted in case of connection errors or server failures, involving various aspects of load balancing. At present, data stored at Caltech, SDSC and Leece is linked by OC-12 Hippi, the vBNS and the Internet. The data consists of sets of SAR channels, with each channel being a file between 5 and 100MB. Channels are monochrome images of the surface of the Earth, with server co-registered channels taken together to form a track.

One of the problems with using the SARA is the quantity of data that could be generated during a single request. If the user requesting analysis on the generated data is not at the data source, hundreds of MB of data may need to be downloaded to the client. There are two ways to deal with this; (1) to download this data to a local parallel computer which is presumably closer to the data source than the user, and perform analysis on the parallel machine, (2) to migrate the computation required by the user to the data source, or to a site that is closer to the data source, perform the required computation, and migrate the results back to the user. At present, approach (1) is adopted, whereby a user must telnet to the center hosting the parallel computer, select computing resources to use, and perform the image processing activity. The results of computation, images or text, are written to web pages and examined with a browser — enabling further data to be imported and processed if desired. The user can also create scripts that will enable production runs that can execute for longer periods of time.

The current approach is very restrictive on the kinds of users that can access and manipulate data in the SARA, as the client may not have permission or the relevant software to analyse data. Also, CGI scripts are stateless, whereby a user cannot enable one request to be based on the outputs or conditions generated from a previous one, unless additional approaches such as cookies, are employed. Furthermore, the output from a CGI script is a MIME-typed data object, which restricts the passing of compound multiple data objects in response to user request. We therefore propose two extensions to the data analysis using SARA, (1) the use of compound objects tagged with XSIL [1], which corresponds to a special purpose tagging format for SARA data, (2) use of mobile agents to carry queries to the data source, rather than moving the data generated in response to a query to the data processing site. In our system, we divide operations performed by a user into a number of interacting agents, with each agent delegated a particular role within the system. In order for these mobile agents to work effectively, we propose a system of interacting agents, where each agent performs a particular role, as described in the next section. Each user is presented with an interface agent, which manages user interactions with the local file system and provides support for launching a query to the SAR data. The query is wrapped as a user request agent, using the Voyager mobile agent library, and delivered to the SARA data server. The agent passes the incoming query to the data server, where the query is executed and any data generated is maintained as a local file. A URL reference of the file is handed back to the user request agent, which can then proceed to another host for additional processing, or to the parent host with the URL of the generated data, and the status of results that are to be sent to the user. The interface agent at the user side receives the results, and displays these to the user.

In case of additional processing, the user request agent is dispatched to additional hosts, and continues to carry the URL for data sources at each intermediate site that it visits. Hence, we do not migrate data sets across networks, as the processing demands may vary from a simple change of data format, to more compute intensive image processing such as principal component analysis or pattern classification.

The multi-agent system

A multi-agent system is currently being developed to undertake data analysis of SARA data. The system contains agents which undertake very specific tasks, and consist of the following types:

i. User Presentation Agent (UPA): this agent supports the user in creating a query/operation to perform on the SARA data set. Typically, the user selects from pre-defined options, or could type a query for the remote data source. A graphical interface is provided, where results of the query are then presented to the user. In a limiting case, these agents can be very simple, and only support the use of pre-defined queries, and a limited support for visualisation. They can also be more complex, and undertake user profiling to automatically categorise users based on a history of queries, generated by the same user, or a group of users at the same site. In the context of SARA, user profiling also helps to re-use results generated from a previous query, and to help users formulate queries based on previous ones. Typical queries can range from the retrieval of an image at a given latitude/longitude, to a more complex query, where a filter is to be applied on an image after retrieval to analyse particular aspect of an image — such as ecological profile of an area.

ii. The query identified by the user, is wrapped using the Voyager library, to a mobile object, which is then migrated to the data source. This we label as a User Request Agent (URA), which carries the wrapped code — SQL or Java code, to the remote site, where it is to be executed. The URA interacts with the UPA and a Local Assistant Agent (LAA) at the remote site. Both the UPA and the LAA are responsible for launching and supporting the URA, and must negotiate security privileges before the URA is launched.

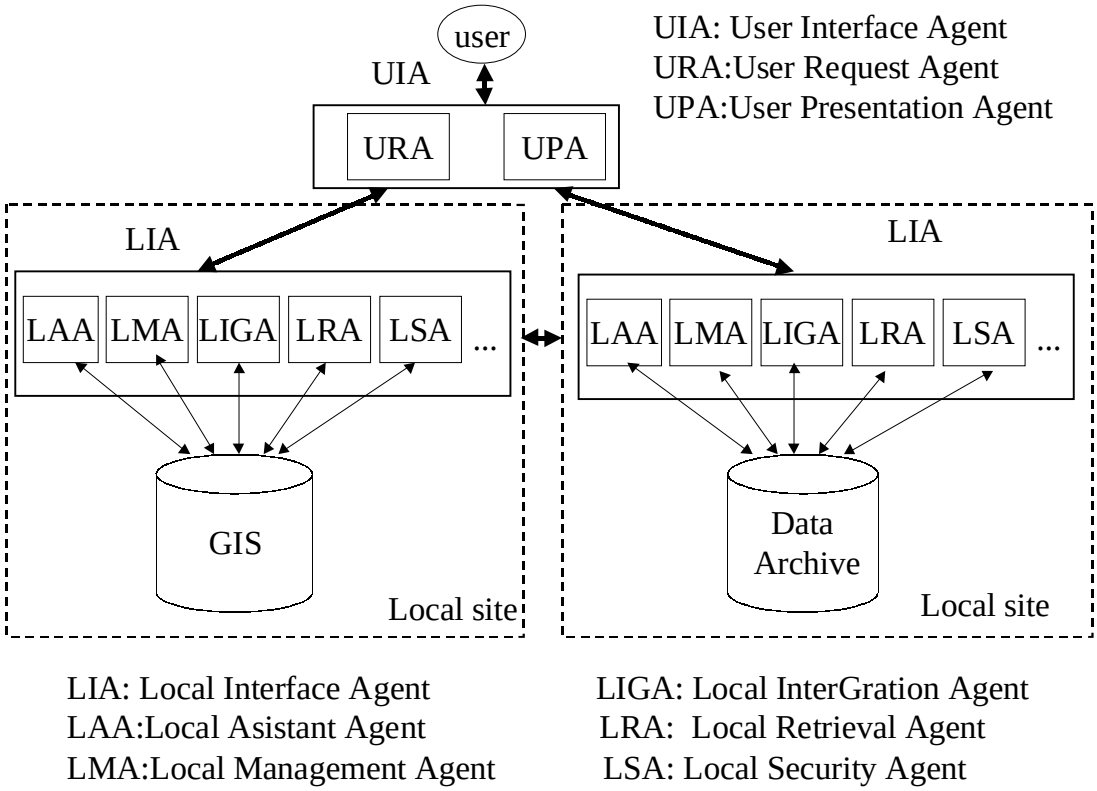

Figure 1: Interaction Between Agents

iii. A Local Assistant Agent (LAA) must be present on each site supporting the SARA archive. The LAA is responsible for receiving the URA, and running the required query on the data source, which can be a structured database or a flat file. The LAA can interact with the Local Management Agent (LMA) at the remote site, to achieve load balancing for instance, if multiple processors are available.

Therefore, we identify three main types of agents: (1) a User Presentation Agent (UPA) (2) a User Request Agent (URA) (mobile), (3) a Local Assistant Agent (LAA). The LAA can interact with other agents, depending on the configuration of the remote site, and can be supported by: (4) a Local Management Agent (LMA), (5) a Local Security Agent, (6) a Local Integration Agent (LIGA). Responses generated by a query are wrapped using XSIL [1], as in code segment below:

```
<track id=13242>
 <name>Almaz, Russia</name>
 <date>04-16-1994</date>
 <width>2824</width>
 <height>8000</height>
 <area>
  <quadrilateral>
   <lon>47.425</lon><lat>45.669</lat>
   <lon>47.703</lon><lat>45.418</lat>
   <lon>48.719</lon><lat>45.963</lat>
   <lon>48.444</lon><lat>46.217</lat>
  </quadrilateral>
 </area>
 <server name=CACR_HPSS>
  <baseurl>http://hpss.cacr.caltech.edu/
...</baseurl>
  <timeout>660</timeout>
 </server>
 ...
</track>
```

Collectively, these agents work to overcome deficiencies found in the current SARA systems, which makes use of CGI scripts. Each agent interacts via tagged messages, and has a particular, well defined role. Figure 1 shows the overall interaction between different agents. Additional information can be found in [3].

A multi-agent system can provide specialised support for managing large data sets, and the role of mobility is of particular benefit here. If the UPA identifies results of previous queries that may be of interest, then the itinerary of a mobile agent can be modified based on results generated from intermediate queries. More importantly, the agent paradigm supports integration of different data formats, and computational resources. We currently have a working prototype, which involves a UPA, a URA and an LAA. Our current work focuses on linking these with other types of services at the remote site. A future goal is to investigate effects on system performance when a large number of users simultaneously request queries on a data source, and to evaluate wrapping granularities for queries generated using the UPA.

References

[1] K. Blackburn, A. Lazzarini, T. Prince and R. Williams. XSIL: Extensible Scientific Interchange Language. In *Proceedings of High Performance Computing and Networks Amsterdam,* pp 513-524, February 1999.

[2] “Voyager”, see web site at: <http://www.objectspace.com/>

[3] Yanyan Yang, Omer F. Rana, David W. Walker, Roy Williams. A Mobile Agent-Based Architecture for On-Demand Processing of Remote-Sensing Archive. In *Proceedings on High Performance Computing Asia*, May 2000 (to appear).

Market-Based Decentralised Process Management using Multi-agent Systems

Torsten Eymann[1] and Paul J. Kearney[2]
[1]Albert-Ludwigs-Universität Freiburg, Germany
[2]BT Laboratories, Ipswich, United Kingdom
[1]eymann@iig.uni-freiburg.de
[2]paul.3.kearney@bt.com

This article is a result of a visit by Torsten Eymann from the University of Freiburg, Germany, to British Telecom Labs, Adastral Park, UK, in February 2000. The visit was funded by a Travel Fellowship Grant from Agentlink[1].

AgentLink brought us together. In 1998 and 1999, the authors met during several AgentLink SIG meetings and discovered a mutual interest in the same research area of market-based process management. Both our research projects, DYNAMO and AVALANCHE[2], obviously shared the same conceptual background, which had been developed independently for the particular project. Unsurprisingly, our technical approaches and implementations have similarities and differences. We wanted to understand these similarities and differences better and explore potential synergies in combining them, but we needed more time together. AgentLink's Travel Fellowship Grant provided the framework to seize that opportunity during two weeks in February 2000.

In this short article, we first describe our common conceptual background. After a short description of the two projects DYNAMO and AVALANCHE, we describe some results of the research visit and one of our possible joint research paths for future work.

Market-based multi-agent systems

There is widespread interest in market mechanisms as a means of co-ordinating the activities of autonomous entities, without rigid, centrally defined organisations. The agents in question may represent companies operating within national or global economies, customers and suppliers in a supply chain, organisational units within an internal market, human or software traders in electronic commerce systems, or software agents managing a distributed materials flow process. Some of the early approaches to market mechanisms have been outlined in the article collections edited by Huberman (1988, cf. agoric open systems) and Clearwater (1996, cf. market-oriented programming). These approaches mainly implement a centralised problem solver, which is able to compute a timesliced resource allocation in a closed, low-populated, and restricted environment.

In contrast to that, the authors hold that the real world application domains envisioned are of a large scale, open to new participants and allow different development paths. Agents in such environments operate in real time and are governed by different independent principals. In our view, the properties of these domains render any centralised approach to coordination useless. Both projects, *AVALANCHE* and *DYNAMO*, advocate therefore a decentralised, market-based approach as a means of harnessing the self-seeking drives of autonomous agents to achieve collective benefit.

The DYNAMO project

The *DYNAMO* (Dynamics of Adaptive Market-based Organisations) team in BT's Intelligent Business Systems Research (IBSR) Group is working to develop an understanding for these concepts through computational experiments and mathematical analysis. Our intention is to express this knowledge as engineering principles, methods and techniques for the specification of decentralised systems that are robust, adaptive and scaleable. In particular, we are interested in systems in which the agents interact by buying and selling services, and hence form a market (Kearney 1999).

A particular family of scenario studied is of parallel 'supply chains' of agents each of which shares a pool of workers and can be considered a composite agent. All chains buy from a single supplier and sell to a single customer each of which has constant bidding behaviour. In a single two-agent supply chain, the first agent consumes the input item type bought from the supplier, to produce an intermediate type, which is consumed by the second agent to produce the type of item sought by the external customer. The two agents are now linked not by a competitive relationship, but one of mutual interests. Neither agent can sustain a profit on it own. Each agent is now operating within a trading environment that changes in response to its actions. The rules of behaviour and market mechanism result in the two agents co-evolving to a state in which their throughputs are equal and at the highest level at which the target unit profit can be sustained. When there are multiple supply chains, trade in work items can also occur between agents in different supply chains. Whether agents are perceived as competing or cooperating depends on context. With 'diseconomies of scale' (dos) production functions, the most efficient overall configuration for the system to adopt is for workers to be allocated equally to all agents. With 'economies of scale' (eos), however, there are multiple optimal configurations, each of which has only one agent of a given type (i.e. in a given step in the supply chain) active. When production is constrained by the number of workers in each pool, the most efficient eos configurations are those with all the workers of each composite agent concentrated in one of its constituent agents, so that only one agent in each row and column is active. This means that the MAS must pick out a particular process (i.e. a path through the MAS from customer to supplier), which requires apparently

coordinated behaviour among the agents making up the MAS. The only communication between agents is via the local trading interactions, however. In this final state the composite agents are in co-dependent states a long way from their normal operating states. Reaching these states involves coordinated behaviour on the part of the agents, yet such behaviour was not explicitly programmed into their behaviour rules. This is a form of co-adaptive self-organisation albeit a simple one.

Other related BT projects discussed during the visit are:

- A collaborative project of the Dynamo team with R.E. Smith of the University of the West of England has worked on the synergies obtained from combining evolutionary computing techniques with those of market-based multi-agent systems (Smith 1999).
- The Agent-enhanced workflow (AEW) team in the IBSR group researches into the use of software agents in the setting-up of decentralised workflows, the reactive and proactive redistribution of work during process failures, the promotion of interoperability between workflow management systems, and the visualisation and monitoring of decentralised business processes (Shepherdson 1999)
- The starting DIET project is part of the EU's research programme on "Universal Information Ecosystems". The purpose of DIET is the construction of a common software framework for applications of decentralised, self-organising information systems.
- The successful ZEUS project provides a generic agent architecture, which can also be used in constructing market-based multi-agent systems (Nwana 1999). Further developments from ZEUS have led to a sophisticated interface for the formalisation of user preferences in markets, and the specification of automated negotiation of the agents.

The AVALANCHE project

The *AVALANCHE* (Agent-based Value Chain Coordination Experiment) team at the Albert-Ludwigs-Universität Freiburg researches into the coordination of supply chains with autonomous software agents. We have build a prototypical Java-based electronic marketplace, where small businesses (e.g. craftsmen like lumberjacks and carpenters) try to sell and buy goods by configuring a (mobile) agent and sending it to the electronic marketplace to autonomously negotiate. The key difference of *AVALANCHE* to other agent-based marketplaces is the absence of a central arbitrator, mediator or auctioneer. We are instead interested in phenomena of self-organising, emergent co-ordination, achieved through the implementation of strictly decentralised economic concepts. The different agent types form a supply chain; producers sell wood to lumberjacks, lumberjacks sell boards to carpenters, carpenters sell plates to cabinetmakers, cabinetmakers sell tables to consumers. The co-ordination question is thus not only one of a specific marketplace, but of a chain of interconnected markets (Eymann 1998).

Co-ordination in the marketplace is a means for making concessions, and every agent is guided by nothing but its own experiences and expectations. The agents in AVALANCHE have no explicit utility function, but simply try to maximise their capital. In a continuous loop, the agents either negotiate to sell one product to an agent further up the supply chain, or make products from materials, or negotiate to buy one material from other agents. The negotiation among the agents can best be characterised as "bilateral haggling", a sequence of propose and counter-propose messages which eventually leads to a compromise price level at which goods and money are exchanged. The competition between agents of the same type leads to an individual agent strategy, where concession making and greediness is in some sort of balance as a functional result of nothing but the strategies of the other agents in the population.

By varying the initial variables which determine the agents' strategies, we can show in subsequent experimental runs how dominant parameter combinations either succeed or are counterbalanced by co-evolving agents. If all agents are equipped with the same strategy, the price levels of the goods will randomly walk around the initial start prices. But on open markets, a certain amount of new agents/strategies will always arrive on the marketplace which are diverse from those currently in action, and which grasp the evolutionary chance of changing the system in their favour. *AVALANCHE* implements a simplified version of a distributed evolutionary algorithm (Smith 1998), where all agent types are able to co-evolve – a highly dynamic pattern arises, where early dominating strategies are counterbalanced, and agents continually develop successful parameter combinations which allow them to survive in an ever-changing environment. These preliminary findings already show the difference of a self-organising market approach to a centralised system, where one would need to constantly monitor and control the agents to assure that the system will suffer no lock-in situation.

Conclusion and outlook

The results of the visit can be found not only in a greater detailed understanding of each other's work, but more important in the opportunity to identify a) concrete application areas for implementing the concept, and b) missing software engineering tools which hinder such implementation. As concrete application areas, we identified those who implement packet-based routing of materials or information flow through a network of resources or actors. Examples of these are the coordination of supply chains and production networks in Virtual Corporation environments, Agent-enhanced workflow in large and decentralised companies, and the routing of data packets in computer networks. On the software engineering level, the ability to have a detailed look at each other's Java-based implementation proved to be very useful. The common conceptual understanding allowed to concentrate on particular aspects of both projects, and we identified the need for a software engineering framework to denote the necessary elements of agents, protocols and institutions, which could serve as building blocks for market-based process management applications. One of our next steps will thus be the investigation on building a common software framework for decentralised market systems, derived from both and related existing implementations.

In total, the research visit can be regarded as highly successful. By keeping in close touch with each other, we will be able to avoid "inventing the wheel twice" on a conceptual level. Instead, we are able to exploit the synergies which come from implementing different applications out of a shared conceptual understanding, and hopefully a shared software framework. This allows each of us to fine-tune his research to the direct application

area, but with a strong view on transferability of the concepts and maybe even interoperability of the implementations.

Without the support of AgentLink, this collaboration would not have been possible. We thus would strongly urge researchers in similar positions to seize the opportunity to visit other groups on a timescale of several weeks, to investigate if synergies can be exploited and research efforts thus can be maximized. It worked for us.

References

Clearwater. *Market-based control: a paradigm for distributed resource allocation*, ed. Scott Clearwater, World Scientific: Singapore, 1995.

Eymann T., Padovan B., Schoder D. Simulating Value Chain Coordination with Artificial Life Agents. In *Proceedings of the 3rd International Conference on Multi-Agent Systems (ICMAS'98), July 2-8, Paris 1998*, IEEE Computer Society Press: Los Alamitos, CA, 1998, p. 423-424.

Huberman. *The Ecology of Computation*, ed. Huberman B.A., North-Holland: Amsterdam, 1988.

Kearney P.J., Merlat W. Modelling market-based decentralised management systems, *BT Technology Journal*, Vol. 17, No. 4, October 1999. p. 145-156.

Nwana H., Ndumu D., Lee L., Collis, J. ZEUS: A Tool-Kit for Building Distributed Multi-Agent Systems, *Applied Artifical Intelligence Journal*, Vol 13 (1), 1999, p129-186.

Shepherdson J.W., Thompson S.G., Odgers B.R. *Decentralised workflows and software agents, BT Technology Journal*, Vol. 17, No. 4, October 1999. p. 65 - 71.

Smith R.E., Taylor N. A Framework for Evolutionary Computation in Agent-Based Systems, *Proceedings of the 1998 International Conference on Intelligent Systems*, 1998 pp. 221-224.

Smith R.E., Kearney P.J., Merlat W. Evolutionary adaptation in autonomous agent systems – a paradigm for the emerging enterprise, *BT Technology Journal*, Vol. 17, No. 4, October 1999. p. 157 -167.

1 We would like to express our gratitude to the Agentlink Network of Excellence, which provided us with the opportunity to get together during several SIG meetings in 1998-99, and which funded the research visit in 2000.

2 More information on both projects can be found under <http://www.labs.bt.com/projects/ibsr/dynamo.htm>, and <http://www.iig.uni-freiburg.de/telematik/projekte/avalanche/index.htm>, respectively.

Grasshopper 2 - the Next Generation Agent Platform

Thomas Magedanz
IKV++ GmbH, Germany
magedanz@ikv.de

In the beginning of the second quarter of 2000, a new age of agent technology and development of agent-based software solutions has begun with the official launch of Version 2 of the Grasshopper agent platform developed by IKV++ GmbH, Germany [1]. As described in AgentLink News 3/1999 [2], Grasshopper is an open Java-based mobile intelligent agent platform, which is compliance to the both available international agent standards, namely the OMG MASIF and FIPA specifications. The first version of Grasshopper has been released in autumn 1998 with a subsequent release (Version 1.2) in February 1999. Since then Grasshopper has achieved a broad recognition around the globe, as it formed the enabling technology for a variety of international research and development projects in the fields of telecommunications and electronic commerce.

Although Grasshopper has proven to be a suitable agent platform for many application contexts, there was room for both technical and distribution policy improvements. This is now addressed through the release of Grasshopper Version 2! Besides a variety of technical enhancements, such as improved performance, reliability, scalability, and extensibility, full Java 2 support, etc., the major news is that Grasshopper 2 is free! This means that potential customers can instantly start using Grasshopper for the fast development of agent-based software systems and applications by a simple registration at the Grasshopper website [3]. Additionally, customers are invited to join the newly formed Grasshopper Community [4], which is also hosted on the Grasshopper website, offering an open discussion forum among Grasshopper users from all over the world.

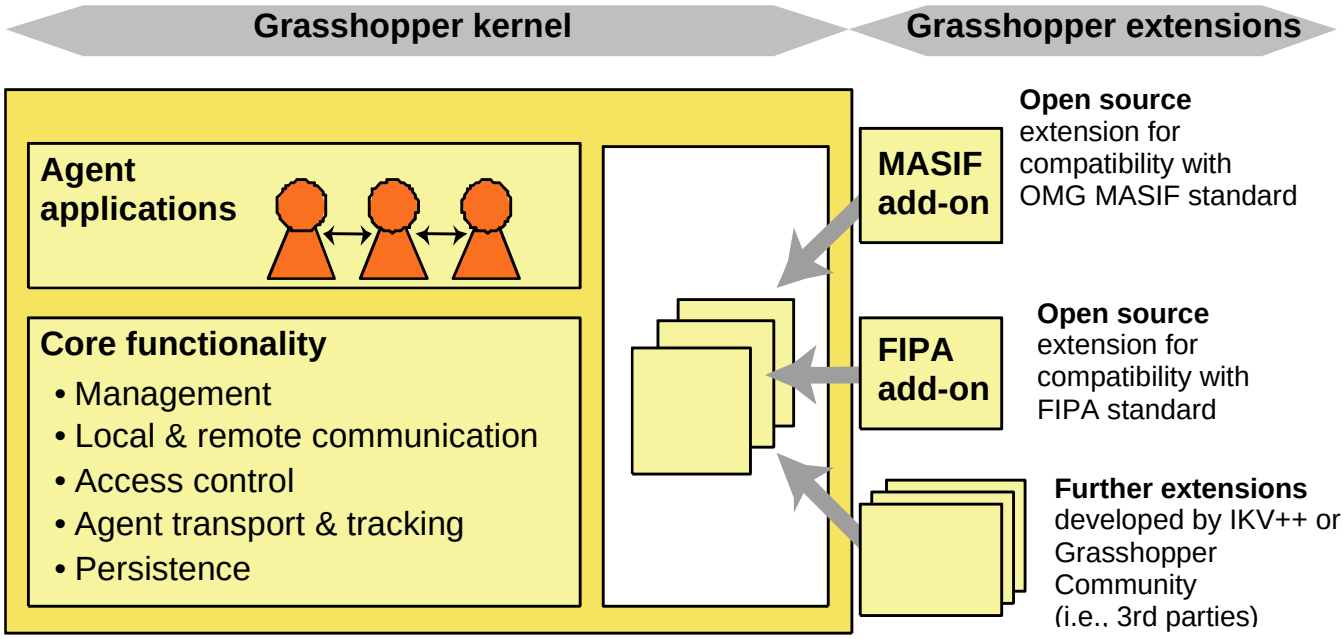

Figure 1: Grasshopper Platform Architecture

Looking at the Grasshopper platform architecture as illustrated in Figure 1, it is now structured into three major parts; a highly tuned core system providing all necessary capabilities for developing and running Grasshopper agents, and two optional open source extensions provide the OMG MASIF and FIPA standard interfaces for agent/platform interoperability. Thus users can modify/enhance these extensions in accord to their individual needs. Further extensions are currently under development at IKV++ are:

- A Grasshopper servlet provides access and control to remote Grasshopper agents and systems from any web browser via the internet.
- Another upcoming extension aiming also for enhanced user interactions, the Grasshopper-Media Framework, provides media converters and telecommunication service interfaces for accessing and delivering Grasshopper agents, enabling the delivering of their contents via phone (Voice), email, fax, SMS and WAP.
- An additional management system providing an SNMP interface will be made available in the course of the year, which makes the administration of Grasshopper and Grasshopper-based application within corporate networks easier.
- Furthermore, it is assumed that additional third party extensions will mushroom within the Grasshopper community.

Grasshopper runs on all operating systems featuring a Java Virtual Machine. Worth mentioning is the upcoming release of an adapted Windows CE version of Grasshopper 2 in order to enable pre-UMTS application developments (for those sharing the view that pocket PCs will be the starting point for next generation UMTS terminals).

For downloading the platform, joining the community or just for obtaining more information about Grasshopper readers should visit the new Grasshopper website: www.grasshopper.de or send an email to info@grasshopper.de. Interested readers will find also valuable information about platform support, training and platform customisation and solution development.

Giving readers an impression of the recent development activities centered around Grasshopper at IKV++, one will find the areas of Mobile Commerce, Virtual Home Environments (UMTS) and new softswitch architectures for converged networks.

References

[1] IKV++ GmbH homepage: <http://www.ikv.de>

[2] M. Höft, T. Magedanz, J. Quantz. Grasshopper - the Agent Platform. In *AgentLink News 3*, July 1999

[3] Grasshopper homepage: <http://www.grasshopper.de>

[4] Grasshopper community homepage: <http://www.grasshopper.de/community>

Cooperative Information Agents Workshop (CIA'99)

Matthias Klusch
DFKI GmbH, Saarbruecken, Germany
klusch@dfki.de

Modern information environments consist mainly of large, distributed and heterogeneous resources connected via Internet, Intranets, or virtual private networks. These environments are open and may dynamically change. The effective search for and management of information as well as the efficient, individual visualization of available information spaces and handling of uncertain, incomplete or vague data are main goals of intelligent information agents on the Internet and Web. To accomplish these tasks agents have to apply methods and techniques especially for reliable human-agent interaction, intelligent integration of different types of information and service mediation via matchmaking or brokering. In addition, the research area of collaboratively adapting and mobile agents for information management on the Internet still is a relatively uncharted territory in information agent technology.

The international workshop series CIA covers the whole thematic range of intelligent and cooperative information agents. The third workshop CIA'99 has been held at the University of Uppsala, Sweden, from July 31 until August 2, 1999, and attracted nearly 50 people to participate in this event. It has been built on the success of the past CIA worskhops and focused on mobile information discovery, advances in collaboration, mediation and negotiation, as well as personal assistance and interaction. The workshop featured 17 regular papers selected out of 47 submissions, and 9 invited lectures from well-known researchers and experts in the field coming from both, the academic and industrial area. The proceedings of CIA'99 has been published in the Springer LNAI series, volume 1652 (proceedings of CIA'98 and CIA'97 as volumes LNAI 1435 and 1202, respectively). The workshop was supported by the co-sponsors DaimlerChrysler AG (Germany), George Mason University (USA), Active Online Systems Ltd. (UK) and the European Network of Excellence for agent-based computing (AgentLink).

Participants have been offered a packed two and a half days scientific program structured into 8 sessions, and a nice boat trip on a nearby beautiful lake with a delicious dinner on board as a common social event. The first day of the workshop was devoted to sessions coping with advances in agent-based information discovery and management, prototypes, systems and applications, and issues of communication and collaboration among autonomous information agents. The workshop started off with two invited talks given by *Walt Trutzkowski* from NASA Goddard Space Flight Center (USA) on agent technology from a NASA perspective, and *Larry Kerschberg* from George Mason University (USA) on resource management in agent-based, distributed information environments, respectively. In particular, the talk by Walt gave exciting insights of NASA's perspective of how to apply information agents technology to accomplish comprehensive, automated collaboration among autonomous ground and space-based exploration systems, such as the ground operation system LOGOS and the spacecraft-oriented Deep Space One Remote Agent experiment. After the lunch break newest prototypes and systems of information agents for an intelligent Web site in insurance domain, Web repository change monitoring, searching the Web exploiting german texts, and intelligent collaborative filtering were presented in two parallel sessions. In the afternoon session progress on some hard problems and issues of communication and collaboration have been reported and intensively discussed. This included initial, empirical investigations of intention reconciliation in the context of teamwork, a similarity evaluation technique for cooperative problem solving in a group of agents, and a rule-based general computational model for cooperating agent systems in constrained environments. The day ended late with an inspiring invited talk on autonomous search for information in uncertain environments given by *Erol Gelenbe* from University of Florida (USA).

On the second full day of CIA'99 workshop three subsequent sessions have been held dealing with hot topics such as issues of mobile information agents, rational agents for electronic business, and service mediation, brokering and mediation. In his very convincing invited talk *Michael Wellman* (Uni Michigan, USA) surveyed the area of agent-based automated commerce on the Internet and also sketched some visions for the future of agent-based e-business including ongoing research on double auctions, bundles and alternative mediation concepts. The following invited lecture given by *Mike Papazoglou* from InfoLab at Tilburg University (Netherlands) focused on the role of agent technology in the business-to-business segment of the e-commerce market. In the afternoon, the final session for the day provided work on methods for service mediation, brokering and negotiation. During the invited talk given by *Amit Sheth* from Georgia University (USA) concepts of semantic information brokering for digital media and the potential of agent technology for a global information infrastructure have been lively discussed. That included issues such as agents for inter-ontology interpretation, media independent correlation of information, and the cooperation among these specific types of agents to achieve their tasks devoted to intelligent interoperation of heterogeneous information systems. This shall be performed in a more advanced way than provided by, for example, multidatabase or federated database systems. The

following regular talks on arbitration and matchmaking among agents with conflicting interests as well as on integrative negotiations among adaptive agents based on a connectionist approach also attracted high interest and stimulated some discussions.

The last day of the workshop has just one session in the morning solely devoted to recent work on personal, adaptive assistants. This session covered in particular three invited talks by *Pat Langley* from DaimlerChrysler Research in Palo Alto (USA) on an adaptive conversational interface, by *Michael Lewis* from University of Pittsburgh on an in-depth study of results from projects TANDEM and MokSAF dealing with human-agent interaction, and by *Toru Ishida* from Kyoto University (Japan) on current status of the digital city Kyoto. After closing the workshop and having lunch a convenient shuttle service from the workshop venue to the closely located swedish capital Stockholm was provided to enable participation in the opening of the IJCAI-99 conference in time.

In summary, the CIA'99 workshop has been very well received and considered as a success by all participants. Like previous workshops in the series it not only offered participants high-quality research contributions to information agent technology but also a distinguishing sequence of outstanding invited talks by experts from different fields of information systems, AI, multiagent systems, and last but not least inspired some lively, interesting discussions. The local organization team headed by Christian Tschudin from Uppsala University did a great job in making the event a very nice, smooth, comfortable and social one.

The next event in the CIA workshop series is CIA-2000 which takes place from July 7 – 9, 2000, in Boston (USA), in conjunction with the international conference on Multiagent Systems (ICMAS-2000). The proceedings are going to be published in the Springer LNAI series. Again, the workshop will especially offer a bunch of excellent invited talks from experts in different related fields in information agent technology. Latest information on the CIA-2000 workshop are available in the Web at <http://www.dfki.de/~klusch/cia2000.html>, or contact the chair via email klusch@dfki.de.

Conference Report

Workshop of the First UK Special Interest Group on Multi-Agent Systems (UKMAS'98)

Michael Luck[1] and Michael Fisher[2]

[1] University of Warwick, Coventry, UK

[2] Manchester Metropolitan University, Manchester, UK

[1] mikeluck@dcs.warwick.ac.uk, [2] M.Fisher@doc.mmu.ac.uk

The 1998 Workshop of the UK Special Interest Group on Multi-Agent Systems was held at the Manchester Conference Centre on 14th and 15th December, chaired and organised by Michael Fisher of Manchester Metropolitan University, continuing the series of focussed and constructive meetings in this field. After two very successful workshops on the Foundations of Multi-Agent Systems at the University of Warwick in 1996 and 1997, the scope was broadened for 1998 to a wider range of issues concerning all aspects of multi-agent systems. About 50 people attended, representing both industry and academia, and from a variety of relevant disciplines. The aim of the workshop was both to facilitate dissemination of recent research within the multi-agent systems community and also to promote discussion within this often diverse area. Again, the two-day workshop was based around a mixture of invited presentations from Keith Decker of the University of Delaware, USA, and Moshe Tennenholtz of the Technion, Israel, paper presentations and panel discussions. Generously supported by the UK's Engineering and Physical Sciences Research Council (EPSRC), AgentLink and Hewlett Packard Laboratories, the aim was to provide an opportunity for promoting and supporting activity in the research and development of multi-agent systems across academia and industry.

There were three panel sessions, which addressed the pragmatic issue of making money from agents [3], the nature of argumentation and negotiation [2], and the possibility and merit of transferring models of agents between disciplines [1]. All three provided engaging discussions, but in this report, we focus on the other contributions to the workshop through paper presentations and invited talks, which cover a wide range of relevant topics. The structure of the report reflects the organisation of the workshop.

The first day of the workshop began with an invited talk from Moshe Tennenholtz of the Technion, Israel, who discussed the relation between economics and artificial intelligence, which have overlapping interests in some important fundamental issues. While economic models typically deal with the behaviour and interaction of rational agents, artificial intelligence deals with the construction of such agents. In spite of these fundamental connections, there still seems to be a considerable distance between them. Tennenholtz identified two major challenges to address in order to bridge between the related theories: the need to re-consider the theory of (economic) mechanism design in view of its use in computational settings; and the desire to incorporate distributed systems features

into game-theoretic models, and study these new models.

In the first of the paper presentations, Beer described his work with Bench-Capon and Sixsmith at the University of Liverpool on the issues involved in managing dialogues between information agents. He focussed on the problems associated with conversation classes derived from an agent-based distance learning application that involves the use of mediators to provide intelligent management of information flow between multiple agents.

Next, Ghidini of Manchester Metropolitan University and the University of Trento described her work with Serafini on information integration for electronic commerce, in which agents need to exchange information with other agents and to integrate the information obtained from other agents in their own information. The goal of the work is to provide a formal semantics for information integration able to cope with distributed, autonomous, partial, and redundant information. Two examples from an electronic commerce scenario were introduced to illustrate the issues.

In the second of the paper sessions two papers concerned with *rights* were presented. Alonso of the University of York began by describing preliminary work on rights and coordination in multi-agent systems. He introduced some intuitive ideas about basic rights or liberties: how they are understood, their functions and their relations with unconstrained actions in a general model of coordination. Alonso proposed the notion of rights that guide but do not control the behaviour of autonomous agents, as restrictions of actions that allow them enough freedom but still constrain them.

In contrast to Alonso's view of rights as liberties, Norman's work at Queen Mary and Westfield College with Sierra and Jennings offers a view of rights by which they are a means of defining flexible agreements between agents in order for them to act in collaboration. He presented a language in which agents are constrained to act to uphold the rights of others and act in accordance with an agreement to which they are bound. Norman argued that the intuitions captured by his model provide a flexible way of describing agreements between agents, while retaining a notion of joint commitment.

The first day of the workshop ended with an effort to understand the relationship between different disciplines contributing to the field of agent-based systems. Edmonds of Manchester Metropolitan University introduced the notion of *social embeddedness* as a way to distinguish between the engineering perspective on agents as constructing systems that meet certain performance criteria in a reliable way, and the social simulation perspective in acting as models of social agents to increase our understanding of them. Edmonds argued that social embeddedness will need to be a feature of many social simulation models since it has practical consequences for agents within them, but that it may not be practically possible with the engineering perspective.

At the start of the second day, Keith Decker gave an invited presentation on coordinating intelligent agents, which focussed on how to get organisations - multiple software agents and humans - to coordinate their activities when they are working on shared, loosely coupled problems, such as engineering design or information gathering. Decker described some useful representations, including TAEMS (Task Analysis and Environment Modeling System), for annotating an agent's representation of its activities, and some approaches, including GPGP (Generalised Partial Global Planning), for designing coordination mechanisms that are adapted to some particular problem-solving environment. Examples were drawn from various projects in distributed information gathering, distributed hospital patient scheduling, and a Boeing Rotorcraft collaborative design project. Decker's research program is involved in developing intelligent software agents and *organisations* of these agents (including sometimes humans) that can operate in environments where there is a lot of uncertainty about what is happening and where there may be time pressures or deadlines. The agents will in general have many goals, some partially overlapping or conflicting. They are not (and cannot) realistically look for optimal solutions, but instead must satisfice — try to find a solution that is 'good enough' in the time and resources that are available.

The final paper session was started by Schroeder of the City University who began by describing his work with Mora and Alferes, concerned with extending earlier work on argumentation semantics for single agents to a multi-agent setting including both argumentation and cooperation. In this work, inference for multi-agent systems and an algorithm for inference are both defined, and an argumentation protocol sketched and demonstrated with an example implemented using vivid agents.

Finally, van Eijk of Utrecht University described work with de Boer, van der Hoek and Meyer on a programming framework for systems of interacting agents. This work extends previous work in the development of a programming language for interacting agents that is based on the semantically well-founded concurrent programming paradigms of CSP and CCP, in which agents can revise their beliefs, by formalising some basic patterns of interaction between communicating agents.

As the field of agent-based systems continues to expand, and the diversity of research grows, the value of well-focussed and directed, yet informal, workshops like UKMAS'98 becomes more pronounced. The UKMAS'98 workshop was a great success, attracting 50 participants and disseminating leading edge research activity amongst the UK multi-agent systems research community. Indeed the way in which it has engaged communities from both academia and industry is demonstrated by the location and organisation of the next workshop in the series, UKMAS'99, which will be held in Bristol in December 1999, chaired by Chris Preist of Hewlett Packard Labs.

References

Aylett, R, Dautenhahn, K, Doran, J, Luck, M, Moss, S and Tennenholtz, M. Can models of agents be transferred between different areas?. In *The Knowledge Engineering Review*, to appear 1999.

Beer, M, d'Inverno, M, Luck, M, Jennings, N, Preist, C and Schroeder, M. Negotiation in Multi-Agent Systems. In *The Knowledge Engineering Review*, to appear 1999.

Cliff, D, Muller, J, Preist, C and Wooldridge, M. Making Money from Agents. In *The Knowledge Engineering Review*, to appear 1999.

7e Journées Francophones d'Intelligence Artificielle Distribuée et des Systèmes Multi-Agents (JFIADSMA'99)

Marie Pierre Gleizes
IRIT, Université Paul Sabatier, Toulouse, France
gleizes@irit.fr

The seventh french speaking workshop on Distributed Artificial Intelligence and Multi-Agent was held in Saint-Gilles, Ile de la Réunion, from 8th to 10th of November 1999. I thank a lot Pierre Marcenac and his team for organising so well the conference. The workshop had 72 participants from France, Belgium, Spain and Switzerland. 50 papers were submitted, 20 of these were accepted as full papers, and 8 as posters. The main area of this workshop concerned Multi-Agent Systems Engineering and applications. Actually, designing a Multi-Agent System (MAS) implies a know-how mainly available only among specialists of the field. In order to be usable by any computer engineer, the methods, models and tools for design support must be developed. These preoccupations are pointed out by many papers of the JFIADSMA'99 conference.

Validation techniques of a MAS

When the model of a system is implemented, validation techniques must be employed in order to verify its adequacy. Barreteau and Bousquet propose an approach based on role playing games. Two application domains were successfully tested. Generally speaking, a developer must be sure that the role playing game is itself well evaluated before beginning the multi-agent system model evaluation.

The observation of a system composed of numerous interacting agents is a difficult problem due to the large set of temporal events and the agents' heterogeneity. Simonin, Coulondre and Ferber store these interrelated events in semi-structured data. This approach allows a temporal analysis of the role of agents in an off-line process.

Modelling organisation

As member of a collective system, an agent is basically member of some group and play roles inside these groups. The Aalaadin system is a software design tool centred on this organisational view. Coming from this organisational view, meta-models can be designed in which the system is not only agent-centred but also group-centred. Gutknecht and Ferber develop a methodology integrating many models as a guide for designing MAS. In the same global approach viewing an agent as a member of an organisation, Hannoun, Boissier, Sichman and Sayettat develop the model Moise. They want to verify the adequacy between the defined organisation and the problem to solve, as well as a way for an agent to reason about itself. The system Antigone developed by Ferraris and Maurel contains primitives to specify joint activities of groups inside a MAS.

A system as interacting components

Actually, studies on MAS are more centred on interactions between agents rather than on their own skills. In this way, Sansonnet gives an approach for reflexive agents to deduce collective behaviours based on local acts, thanks to the ability to compute them on chronicles. Observing interactions in a communication and information system, allows to verify the adequacy among the multiple viewpoints of agents involved in a global task. The ABIS platform developed by Durand, Lesage, Cardon and Tranouez allows this observation from different organisational approaches.

A MAS may have to solve many global tasks involving many different groups during time. Thus, groups must be created on demand: this is the basic approach of the contract net protocol. Aknine and Pinson improve the allocation protocol in allowing multi-tasking and avoiding deadlocks. Another way to verify the adequacy between the communication language and the objective of the MAS, consists in the observation protocols during agent's interaction. This is the approach chosen by El Fallah-Seghrouchni and Mazouzi in using colored Petri nets.

Interactions can also occur among intricate levels of organisations. Mezura, Ocello, Demazeau and Baejs propose a "recursive" approach for designing MAS, where an agent having a function, tasks and skills can be composed of simpler agents. The "recursive engine" is a tool to statically design organisation levels of a whole system. It also allows the interaction capabilities between those levels while the system is running. A holonic approach simultaneously considers an agent, as a member of a whole system and as an autonomous entity. Adam, Mandiau and Kolski applied it in SOHTCO, a design support system for co-operative work.

Learning and emergence

In dynamic environments, the specification of MAS cannot be complete and the system must adapt itself autonomously. This ability is implemented in a self-organising approach developed in Carré, Machonin and Glize's work. Thus, the global function emerges from local agents activities, which are fully unaware of it. Emergence is also analysed by Quinqueton and Hamadi helped by a biological inspiration. A prospective point of view dealing with adaptation is studied for collective robotics by Drogoul and Picault pointing out the problem of sociality.

Work on agents

Guillement, Haïk, Meurisse, Briot and Lhuillier consider an agent having a set of basic behaviours components. This leads to design a methodology for an isolated agent in which reusability is a masterpiece, similar to more classical approaches used in software engineering. In the Girault and Stinckwich's system an agent must have skills allowing simultaneously reactive and cognitive behaviours. They propose a diagonal hybrid architecture of agent.

Applications

Bourjot, Chevrier, Bernard and Krafft analyse social co-operation based on the modelling of web building by a particular spider specie. A MAS approach can be useful for physical system analysis. Breton, Zucker and Clément have thus obtained results in granular physics modelling where each grain is a simple agent. Dynamic workshop scheduling in a disrupted manufacturing context is also a difficult problem studied by Tranvouez and Espinasse. The car production is another industrial domain approached by Foisel, Drogoul, Cayrol, Attia and Chauvat.

Invited papers

The workshop included also two invited talks. The topics of these conferences were very different and they were appreciated a lot by participants. The first conference was held by Carles Sierra. He draw a complete view of existing systems dedicated to electronic commerce and he described the work of the Agentlink SIG AMEC. The second conference presented by Marco Dorigo showed how and why the collective behaviour of social insects could be a source of inspiration to realise algorithms or optimised methods.

Conference web site: <http://www.univ-reunion.fr/~jf99/>

What's Happening in AgentLink?

Mike Wooldridge
AgentLink, University of Liverpool, Liverpool, UK
M.J.Wooldridge@csc.liv.ac.uk

AgentLink II news - a note from the network coordinator

First, some news about the progress of AgentLink II. The proposal has been formally approved by the European Commission, and we are currently negotiating the contract. Assuming all goes well, the project will be up and running by July 2000 as planned. Please note that we are not allowed to spend AL2 money on events that occur before the start of the project, unfortunately.

Second, I note that it is exactly three years since I first started work on the original AgentLink proposal, and first started contacting potential AgentLink members. Since then, organising the network and establishing it has taken an enormous amount of my time and energy. Most recently, putting together the project proposal for AgentLink II absorbed a several months! Following my recent move to the University of Liverpool, I decided that when AgentLink II began, I would prefer not to be the main coordinator of AgentLink.

The AgentLink management committee, after being informed of my decision, undertook a process of finding a new coordinator. I strongly felt that the new coordinating site should be a member of the AgentLink management comittee, for reasons of continuity. Members of the management committee were invited to put themselves forward as candidates for the post. In the event, only one such node put themselves forward - the University of Southampton (Nick Jennings, Mike Luck), and the management committee subsequently agreed to Southampton being the coordinating node.

I am delighted to have played my role in establishing AgentLink, and I am doubly delighted to have steered AgentLink through to a position where another three years funding now looks to be agreed. It goes without saying that I will continue to play an active part in AgentLink activities and management - not least the second summer school, which will be the first main event in the life of AgentLink II!

As to the mechanics of the changeover, we will of course be doing everything possible to make things as smooth and trouble-free as possible.

Publications and events coordinator to leave AgentLink

Hugo Brailsford, the AgentLink publications, events and administrative coordinator is to leave AgentLink at the end of its first phase. He will be working on AgentLink until the 13th June 2000. To ensure that the handover of his responsibilities is as smooth as possible, all emails sent to coordinator@agentlink.org and postal mail addressed to AgentLink at Queen Mary and Westfield College will be forwarded to the new AgentLink management team at the University of Southampton from that date onward. Hugo will be taking up a position at a public relations consultancy.

AgentLink 1 financial statement available

A financial statement is available, detailing how the AgentLink 1 grant was spent. To obtain the statement (PDF format), go to the WWW page (http://www.agentlink.org/) and follow the link to "document archive" - the relevant document is 2000-001.

In summary, we have spent:

- 100 K Euro on funding six “special interest groups”;
- 60 K Euro on funding members of the network to attend workshops and conferences;
- 50 K euro on providing financial support for conferences and workshops, including the purchase of conference registrations for AL members;
- 33 K Euro on funding a major European summer school, bringing together the very best researchers and teachers from across the world, paying for 40 students from across Europe to attend outright, and providing significant subsidies for 120 other students;
- 12 K Euro for publishing a regular newsletter;
- 6K Euro for providing members with subscriptions to the “Autonomous Agents and Multi-Agent Systems” journal.

No less that *seventy percent* of the original grant AgentLink received has gone directly to members of the network, in the form of these and other activities. The remainder has been spent on essential administrative support activities, such as the provision of the WWW site.

EASSS 2000

The Second European Agent Systems Summer School (EASSS'2000) <http://www.dfki.de/easss/> will be held in Saarbruecken, Germany, August 14 - 18, 2000.

EASSS'2000 is organized by AgentLink and hosted by DFKI GmbH <http://www.dfki.de/> and Universitaet des Saarlandes <http://www.uni-sb.de/>. The school will consist of a mixture of introductory and advanced courses. The courses will be presented by internationally leading experts in the field and cover the full range of theoretical and practical aspects of agent-based computing.

Course topics

- Foundations of Intelligent Agents and Multiagent Systems
- Logical Foundations of Agent Systems
- Problem Solving and Planning
- Intelligent Cooperative Information Systems
- Coordination, Communication, and Collaboration
- Automated Negotiation and Decision Making
- Computational Markets
- Organization Design
- Societies of Artificial Agents and Social Simulation
- Behavior-oriented Control of Physical Agents
- Learning Agents
- Interface Agents
- Mobile Agents and Security
- Agent-oriented Software Engineering
- Intelligent Agents for Telecommunication Applications
- Agent-Mediated Electronic Business
- Industrial and Commercial Applications
- Agents Standards Activities

Furthermore, there will be evening talks on special subjects. Details of the programme will be given on the EASSS'2000 home page at <http://www.dfki.de/easss/>.There will also be a warm-up event on the 15th of August in an informal environment.

Please consult the EASSS 2000 web site for details on how to register. Some financial support is available for full-time PhD students registered at universities that are members of AgentLink. Consult the EASSS 2000 web site for details on how to apply.

Trading Agent Competition at ICMAS-00

Trading agent competition

Trading in electronic markets is increasingly becoming both a commonplace economic activity and a topic of special interest within the AI, Electronic Commerce, and Multi-agent Systems (MAS) research communities. Inspired by the success of competitions in various other AI realms, we announce the ICMAS-00 Trading Agent Competition, to be held in conjunction with a special purpose workshop at the Fourth International Conference on Multi-agent Systems (ICMAS-00), in Boston in July 2000. This event is designed to spur research on common problems, promote definitions of benchmarks and standard problem descriptions, and showcase current technologies.

The competition will pit software agents–developed by research groups, students, and others from all over the world–against each other in a challenging market game. The software agents will represent travel coordinators whose goal is to arrange travel packages for clients. These travel packages consist of flights, hotel rooms, and tickets to entertainment events, all of which the agents buy (and, in the case of event tickets, sell) in electronic auctions. The market game has been specially designed to present agents with difficult decision problems and admit a wide variety of potential bidding strategies.

The competition will be hosted on a version of the Michigan Internet AuctionBot that has been outfitted with features to facilitate and control distributed market games. Software agents communicate with the AuctionBot via a TCP-based agent programming interface. Simple example agents are available in a variety of platforms and programming languages.

The final round of the competition will be held at the ICMAS-00 TAC workshop on 8 July 2000. More information can be found at <http://tac.eecs.umich.edu/>.

2001		
ICAIL 2001	8th International Conference on Artificial Intelligence and Law, St. Louis, USA.	May 21-25, 2001 http://www.cs.wustl.edu/icail2001/

2000		
MAMA 2000	International ICSC Symposium on Multi-Agents and Mobile Agents in Virtual Organizations and E-Commerce, Wollongong, Australia.	December 11-13, 2000 http://www.icsc.ab.ca/150-rel.htm
ASA/MA 2000	2nd Intl. Symposium on Agent Systems and Applications with the 4th International Symposium on Mobile Agents, Zurich, Switzerland.	September 13-15, 2000 http://www.inf.ethz.ch/ASA-MA/
USM 2000	3rd IFIP/GI International Conference on Trends towards a New Service Market, Munich, Germany.	September 12-14, 2000 http://usm2000.informatik.uni-muenchen.de/
SAB 2000	The Sixth Internatinal Conference on the Simulation of Adaptive Behaviour, Paris, France.	September 11-15, 2000 http://www-poleia.lip6.fr/~sab2000/
CoopIS 2000	Fifth IFCIS International Conference on Cooperative Information Systems, Eilat, Israel.	September 6-8, 2000 http://www.haifa.il.ibm.com/coopis2000.html
HoloMAS 2000	Industrial Applications of Holonic and Multi-Agent Systems in conjunction with DEXA 2000, Greenwich, UK.	September 6-8, 2000 http://cyber.felk.cvut.cz/HoloMAS2000
ABIS 2000	International Workshop on Agent-Based Information Systems in conjunction with DEXA 2000, Greenwich, UK.	September 4-8, 2000 http://www.ntu.edu.sg/home/awkng/abis2000.htm
PRIIA 2000	1st Pacific Rim International Workshop on Intelligent Information Agents in conjunction with PRICAI 2000, Melbourne, Australia.	August 29, 2000 http://www.dstc.monash.edu.au/agents/priia2000
PRIMA 2000	3rd Pacific Rim International Workshop on Multi-Agents in conjunction with PRICAI 2000, Melbourne, Australia.	August 28-29, 2000 http://www.lab7.kuis.kyoto-u.ac.jp/prima2000
IMACS 2000	Special Session on Agent-Based Simulation, Planning and Control at 16th IMACS World Congress 2000, Lausanne, Switzerland.	August 21-25, 2000 http://www.sztaki.hu/~vancza/imacs/agents.html
ECAI 2000	14th European Conference on Artificial Intelligence Berlin, Germany.	August 20-25, 2000 http://www.ecai2000.hu-berlin.de/
KBEM'00	The AAAI-2000 Workshop on Knowledge-based Electronic Markets, at the National Conference on Artificial Intelligence, Austin, USA.	31 July, 2000. http://www.igec.umbc.edu/kbem/
AOIS 2000	2nd International Bi-Conference Workshop on Agent-Oriented Information Systems held at AAAI-2000, Austin, USA.	July 30 , 2000. http://www.AOIS.org
SOMAS	Self-organisation in Multi-agent Systems, Milton Keynes, UK.	July 27-28 , 2000. http://images.ee.umist.ac.uk/emergent/
CLIMA-00	Workshop on Computational Logic in Multi-Agent Systems held at CL-2000 London, UK.	July 24-29 , 2000. http://mhjcc3-ei.eng.hokudai.ac.jp/clima.html
SCI-2000	Session on Multi-Agent Systems at the 4th World Multiconference on Systems, Cybernetics and Informatics, Orlando, USA.	July 23-26, 2000 http://www-users.cs.york.ac.uk/ea/orlando.html
IMASE 2000	Workshop on Intelligent Multi-agent Systems for E-commerce at GECCO-2000, Las Vegas, USA.	8-14 July, 2000 http://www.cwi.nl/~bill/imase
ICMAS 2000	Fourth International Conference on Multiagent Systems Boston, USA.	July 7-12, 2000 http://www.icmas.lania.mx/
ATAL 2000	Seventh International Workshop on Agent Theories, Architectures and Languages, Boston, USA.	July 7-9, 2000 http://www.atal.org/
WET ICE 2000	IEEE Eighth International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Maryland, USA.	June 14-16, 2000. http://www.ida.liu.se/conferences/WETICE/WETICE2000/
AOSE-2000	Agent-Oriented Software Engineering held at the 22nd International Conference on Software Engineering (ICSE-2000), Limerick, Ireland.	June 10, 2000. http://www.csc.liv.ac.uk/~mjw/aose2000/
AOIS 2000	2nd International Bi-Conference Workshop on Agent-Oriented Information Systems, at CAiSE*00, Stockholm, Sweden.	June 5-6, 2000. http://www.AOIS.org

About *AgentLink News*

The aim of the AgentLink newsletter is to provide a relatively informal way of communicating both what's happening in AgentLink, but also what's happening in the agent world generally. Many newsletters end up being rather dull. (Let's face it, the very name "newsletter" puts many people off.) AgentLink News aims to be different. In addition to containing the worthy-but-dull details of what's happening in the network, we aim to carry a range of articles including features, reports on conferences and workshops, informal descriptions of research results and new software, book reviews, and so on. Of course, we can't do this without your help! We need you to generate the content for the newsletter. If you are interested in writing something for the newsletter, please get in touch with the editor Paul Davidsson directly, at the address below. The deadline for receipt of articles for issue six is 15th September 2000. Remember: AgentLink news is not an academic journal, so we won't publish academic articles. Pieces should follow the conventions of similar sorts of publications (such as AI Magazine, or IEEE Internet), and be relaxed in style, with short lists of references, etc. Length is also an issue – features should be no more than a few pages.

The newsletter will have a circulation of up to several thousand. It's an ideal way to communicate your work to a specialist community, who will want to hear about it. So why not contribute?

How to Contact AgentLink

AgentLink general coordinator
Michael Wooldridge
Department of Computer Science
University of Liverpool
Chadwick Building, Peach Street
Liverpool L69 7ZF, UK
Email: coordinator@agentlink.org

AgentLink administrator & WWW manager
Hugo Brailsford
Department of Electronic Engineering
Queen Mary and Westfield College
University of London, London E1 4NS
United Kingdom
Email: coordinator@agentlink.org

Industrial Action (workpackage 1) coordinator
Joerg P. Mueller
Siemens AG
ZT IK 6
D-81730 Munich
Germany
Email: joerg.mueller@mchp.siemens.de

Research (workpackage 2) coordinator
Yves Demazeau
Laboratoire LEIBNIZ - Institut IMAG
46, avenue Felix Viallet
38000 Grenoble
France
Email: Yves.Demazeau@imag.fr

Teaching & Training (workpackage 3) coordinator
Gerhard Weiss
Institut fuer Informatik
Technisch Universitaet Muenchen
D-80290 Munich
Germany
Email: weissg@informatik.tu-muenchen.de

AgentLink News Editor
Paul Davidsson
Department of Computer Science
University of Karlskrona/Ronneby
372 25 Ronneby
Sweden
Email: Paul.Davidsson@ipd.hk-r.se

AgentLink News design by Hugo Brailsford