ISSN 1465-3842

AgentLink NEWS 2

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

January 1999 **www.agentlink.org**

Agent-based computing is rapidly maturing from a relatively obscure existence in research labs towards an accepted and valued technology in industry. A sure sign of this transformation is the emergence of commercial tools for the development of agent-based software. In this issue we feature two articles that describe such tools. Although they both have a strong relationship to the Java programming language, they have adopted very different approaches.

ZEUS, from British Telecom's Martlesham Heath Labs, is an experimental toolkit for the entire agent system development process. It makes good use of graphical programming, debugging, and visualisation tools, and sets a high standard for integrated, GUI-based agent-based development environments.

By contrast, the JACK language from Agent-Oriented Pty in Australia, is a commercial language-based agent development system. It extends the Java language itself, for example by adding agents and plans as first class components of the language. JACK is already in use for a number of industrial projects, and has a number of extensions for, amongst other things, BDI system development.

As well as these feature articles, this issue also includes reports from the three AgentLink Special Interest Groups (SIGs) that had their first meetings in Brussels on September 24, 1998. It is not an exaggeration to say that the SIGs constitute some of the most important concerns of AgentLink. For the uninitiated, there is an introductory text explaining the rationale behind SIGs.

The newsletter is, of course, also filled with interesting conference, project, and site reports. For instance, you will find a report from CLIMATE, a pool of EC-funded (ACTS) projects in the area of intelligent mobile agents for telecommunication applications. CLIMATE also serves as an illustration of how strategically important the European Union regards research and development in the area of agent-based computing.

Paul Davidsson, Editor.

JACK Intelligent Agents - Components for Intelligent Agents in Java

Paolo Busetta, Ralph Rönnquist, Andrew Hodgson, and Andrew Lucas
Agent Oriented Software Pty. Ltd., Melbourne, Australia
{paolo,ralph,ash,cal}@agent-software.com.au

Intelligent Agents are being used for modelling simple rational behaviours in a wide range of distributed applications. In particular, multi-agent architectures based on the Belief-Desire-Intention model have been used successfully in situations where modelling of human reasoning and team behaviour are needed, such as simulating tactical decision-making in air operations and command and control structures. Other applications include intelligent decision support, telephone call centres, and air traffic management.

The JACK Intelligent Agents™ framework by Agent Oriented Software brings the concept of intelligent agents into the mainstream of commercial software engineering and Java. JACK Intelligent Agents is a third generation agent framework, designed as a set of light-weight components with high performance and strong data typing.

We present the design approach and major technical characteristics of JACK Intelligent Agents. An outline of a typical development process involving the framework is given. Also, we discuss the benefits of the component-based approach, both for the software engineer developing sophisticated distributed applications, and for the researcher exploring agent models and architectures.

Introduction

Intelligent Agents are being used for modelling simple rational behaviours in a wide range of distributed applications. Intelligent agents have received various, if not contradictory, definitions; by general consensus, they must show some degree of autonomy, social ability, and combine pro-active and reactive behaviour [5]. One of the better known and most successful architectures for agents is the so-called BDI (Belief-Desire-Intention) architecture, which has seen a number of academic and industrial applications.

Agent Oriented Software Pty. Ltd. (AOS), based in Melbourne, Australia, has built JACK Intelligent Agents™, a framework in Java for multi-agent system development. The company's aim is to provide a platform for commercial, industrial and research applications. To this end, its framework supplies a high performance, light-weight implementation of the BDI architecture, and can be easily extended to support different agent models or specific application requirements.

This paper is organised as follows. Section 2 introduces JACK Intelligent Agents, presenting the approach taken by AOS to its design and outlining its major engineering characteristics. The BDI model is discussed briefly in Section 3. Section 4 gives an outline of how to build an application with JACK Intelligent Agents. Finally, in Section 5 we discuss how the use of this framework can be beneficial to both engineers and researchers. For brevity, we will refer to JACK Intelligent Agents simply as "JACK".

The JACK Approach

Major design goals for JACK were: to provide developers with a robust, stable, light-weight product; to satisfy a variety of practical application needs; to ease technology transfer from research to industry; and to enable further applied research. It has been designed for extension by properly trained engineers, familiar with agent concepts and with a sound understanding of concurrent object-oriented programming.

Whilst applications can be built from the ground up adopting an agent oriented methodology and an appropriate framework, most organisations already possess and depend upon large legacy software systems. Thus, JACK agents have been designed mainly for use as components of larger environments. Consequently, an agent must coexist and be visible as simply another object by non-agent software. Conversely, a JACK programmer must be allowed to easily access any other component of a system. Type safeness when accessing data, reliability and support for a proper engineering process are then key requirements in this kind of environment.

For similar reasons, JACK agents are not bound to any specific agent communications language. Nothing prevents the adoption of high-level symbolic protocol a la KQML, possibly by integrating software already existing in the public domain. However, JACK has been geared towards industrial object-oriented middleware (such as CORBA) and message passing infrastructures (for instance, PVM or DIS in simulation environments). In addition, JACK provides a native light-weight communications infrastructure for situations where high performance is required.

Overview of the JACK framework

From an engineering perspective, JACK consists of architecture-independent facilities, plus a set of *plug-in* components that address the requirements of specific agent architectures. The plug-ins supplied with version 1.2, released at the end of October 1998, include support for the BDI model.

To an application programmer, JACK currently consists of three main extensions to Java. The first is a set of syntac-

tical additions to its host language. These additions, in turn, can be divided as follows:

- a small number of keywords for the identification of the main components of an agent (such as *agent*, *plan* and *event*);
- a set of statements for the declaration of attributes and other characteristics of the components (for instance, the information contained in beliefs or carried by events). All attributes are strongly typed;
- a set of statements for the definition of static relationships (for instance, which plans can be adopted to react to a certain event);
- a set of statements for the manipulation of an agent's state (for instance, additions of new goals or sub-goals to be achieved, changes of beliefs, interaction with other agents).

Furthermore, the programmer can use Java statements within the components of an agent.

For the convenience of programmers, in particular those with a background in AI, JACK also supports logical variables and cursors. These are particularly helpful when querying the state of an agent's beliefs. Their semantics is mid-way between logic programming languages (with the addition of type checking Java style) and embedded SQL.

The second extension to Java is a compiler that converts the syntactic additions described above into pure Java classes and statements that can be loaded with, and be called by, other Java code. The compiler also partially transforms the code of plans in order to obtain the correct semantics of the BDI architecture.

Finally, a set of classes (called the *kernel*) provides the required run-time support to the generated code. This includes:

- the automatic management of concurrency among tasks being pursued in parallel (*intentions* in the BDI terminology);
- the default behaviour of the agent in reaction to events, failure of actions and tasks, and so on; and
- a native light-weight, high performance communications infrastructure for multi-agent applications.

Importantly, the JACK kernel supports multiple agents within a single process. This is particularly convenient for saving system resources. For instance, agents that perform only short computations or share most of their code or data can be grouped together.

A JACK programmer can extend or change the architecture of an agent by providing new plug-ins. In most cases, this simply means to override the default Java methods provided by the kernel or supply new classes for run-time support. However, it is possible to add further syntactic extensions to be handled by the JACK compiler. Similarly, a different communications infrastructure can be supplied by overriding the appropriate run-time methods. Future versions of JACK will extend the base BDI model with new plug-ins and will add a number of development and monitoring tools.

Belief-Desire-Intention Agents

The BDI agent model supported by JACK v1.2 has its roots in philosophy and cognitive science, and in particular in the work of Bratman on rational agents [1]. A rational agent has bounded resources, limited understanding and incomplete knowledge of what happens in the environment it lives. Such an agent has *beliefs* about the world and *desires* to satisfy, driving it to form *intentions* to act. An intention is a commitment to perform a *plan*. In general, a plan is only partially specified at the time of its formulation since the exact steps to be performed may depend on the state of the environment when they are eventually executed. The activity of a rational agent consists of performing the actions that it intended to execute without any further reasoning, until it is forced to a revision of its own intentions by changes to its beliefs or desires. Beliefs, desires and intentions are called *mental attitudes* (or mental states) of an agent.

Observe that BDI agents depart from purely deductive systems and other traditional AI models because of the concept of intentionality, which significantly reduces the extent of deliberation required. BDI has demonstrated to be well suited to modelling certain types of behaviour, such as the application of standard operational procedures by trained staff. It has been successfully adopted in fields as diverse as simulation of military tactics, application of business rules in workflows, and diagnostics in telecommunication networks.

Based on previous research and practical application, Rao and Georgeff [4] have described a computational model for a generic software system implementing a BDI agent. Such a system is an example of event-driven programs. In reaction to an event, for instance a change in the environment or its own beliefs, a BDI agent adopts a *plan* as one of its intentions. Plans are precompiled procedures that depend on a set of conditions for being applicable. The process of adopting a plan as one of the agent's intentions may require a selection among multiple candidates.

The agent executes the steps of the plans that it has adopted as intentions until further deliberation is required; this may happen because of new events or the failure or successful conclusion of existing intentions. A step of a plan can consist of adding a *goal* (that is, a desire to achieve a certain objective) to the agent itself, changing its beliefs, interacting with other agents, and any other atomic action on the agent's own state or the external world.

The abstract BDI architecture has been implemented in a number of systems. Of these, two are of particular relevance to JACK since they represent its immediate predecessors. The first generation is typified by the Procedural Reasoning System (PRS) [2], developed by SRI International in the mid '80s. dMARS [3], built in the mid '90s by the Australian Artificial Intelligence Institute in Melbourne, Australia, is a second generation system. dMARS has been used as development platform for a number of technology demonstrator applications, including simulations of tactical decision-making in air operations and air traffic management.

Application development with JACK

In an ideal setting, a developer building an application with JACK should start by identifying the distributed components of the system. The design of a multi-agent application requires a sound understanding of distributed system development and distributed AI principles that we cannot discuss here. However, observe that in practical situations the decision as to how to distribute functionality may be dictated by a number of external con-

straints, such as the existence of legacy systems or a specific communications infrastructure.

For this discussion, let us assume that the functionality that has to be provided by an agent has been identified and that the BDI model has been chosen. At this stage, two main activities that have to be performed are (not necessarily in the order given below):

- identifying the elementary classes (that is, abstract data types and the operation allowed on them) that are required to manipulate the resources used by the agent. These could be external (relational databases, the Internet, the arms of a robot, a GUI and so on) as well as internal (for instance, specific mathematical data structures to represent financial or spatial information);
- identifying those elements that constitute the mental states of the agent. This boils down to finding:
  - which external events drive the agent (including messages from other agents);
  - which goals the agent can set for itself;
  - which beliefs influence the adoption of plans; and finally
  - the procedures (that is, the plans in BDI terms) required to accomplish tasks, achieve goals and react to events in the various possible contexts.

The implementation of an agent is then a mix of normal Java code for the elementary classes and extended Java for the agent-specific components. The plans of a JACK agent are, in general, sequences of operations on elementary objects, manipulations of the mental states (e.g., submitting sub-goals or changing beliefs) and interactions with other agents.

Observe that a JACK plan could be represented as procedural logic in a *flow diagram*, *state diagram*, *coordination diagram* or other similar notations in an object-oriented methodology such as UML. This is to say that JACK could be used as an extended object-oriented framework supporting event driven and procedural logic in a concurrent execution environment, with the additional benefits of sensitivity to the context and sophisticated management of failure provided by the BDI architecture.

An example

To give a sample of the code of a JACK agent, we have extracted an example from one of the tutorials that are part of the version 1.2 user manual. The purpose is not to show agent programming but to illustrate how JACK code looks as a straightforward Java extension. This section can be passed over by those not familiar with Java.

This example has agents that "ping" each other, that is, exchange empty messages. The message being exchanged is represented as an *event* which is originated by an agent and notified to another:

```
event PingEvent extends MessageEvent {
    int value;


     #posted as ping(int value)
     {
       this.value = value;
     }
   }
```

Note the "event" keyword, in place of "class" in Java. The event is also declared "MessageEvent", which means that it can be notified to another agent; this drives JACK to bring in all the required communications support. The *#posted as* statement declares how the event is generated (in this case, by invoking a Java method "ping()" with one integer parameter).

The following plan handles the notification of the event above and replies to its sender by "bouncing" the event back. This simple example is not sensitive to the context, i.e., there is no restriction on the state of the beliefs of the agent for its applicability.

```
plan BouncingPlan extends Plan {

  #handles event PingEvent pev;
  #sends event PingEvent pev;


     body()
     {
@send ( ev.from, pev.ping(ev.value + 1) );
 /// Reply to the sender of the event
     }
   }
```

A trivial "ping agent" is defined below. It has a single plan and handles a single event. When its method "ping (String other)" is called, it notifies the PingEvent with value 1 to the agent called "other". If the latter is another PingAgent, then BouncingPlan above is invoked, a PingEvent with value 2 is sent back, and so on to infinity.

```
agent PingAgent extends Agent {

     #handles event PingEvent;
     #uses plan PingPlan;

     #posts event PingEvent pev;

     void ping (String other)
     {
       send(other, pev.ping(1));
     }
   }
```

The application can instantiate as many PingAgent agents it desires. The name of the agents and their network addresses are determined by the communications mechanism in use; as said before, JACK provides a high performance messaging system with a simple naming scheme.

Benefits of JACK

The approach taken by JACK has a number of advantages in comparison with both other agent frameworks coming from the artificial intelligence world and standard object-oriented architectures.

The adoption of Java guarantees a widely available, well-supported execution environment. In addition to the promises of the language (summarised by the well known slogan "compile once, run everywhere" by Sun Microsystems), we expect that an increasing number of software components, tools and trained engineers will be available in the next few years.

To the AI researcher, the adoption of an imperative, relatively low-level language such as Java means losing some of the expressive power offered by frameworks based on logic or functional languages. However this is compensated, not only by the universal availability mentioned above, but also by the modular approach of JACK. As said in the previous sections, most components of the framework can be tuned and tailored. This makes JACK particularly suited to experimentation with new agent architectures in order to try outnew functionality (new mental attitudes, different semantics, additional types of knowledge bases, and so on) or to study performance characteristics in specific contexts.

Moreover, when compared with frameworks based on traditional AI languages, JACK has distinctive advantages due to a proper utilisation of the intrinsic charac-

teristics of Java. The most important is strong typing, which reduces the chances of programming errors introduced by simple mis-typing. It also provides a very basic version control by making sure that interfaces are compatible at run-time. Next is performance, which makes the execution speed of agent code written in JACK comparable to a direct implementation in C or C++.

For the engineer developing a sophisticated distributed application, JACK offers several interesting aspects; for instance:

- an efficient way to express high level procedural logic within an object-oriented environment.

  This also helps in rapid application development by allowing a clear distinction between abstract data types and their operations on the one side and, on the other side, application-specific behaviour requiring fine-tuning or evolution when the system is already operational. While the former should be based on high performance, well tested, highly reusable and ultimately expensive code, the latter is better expressed as plans which can be easily modified;
- the context sensitivity and sophisticated semantics of mental attitudes of the BDI architecture.

  This characteristics enable some levels of adaptability to changing conditions;
- ease of integration with legacy systems.

  This enables, among other things, an incremental approach to distributed system development.

When compared with frameworks originating from research environments, JACK has the clear advantages of being light-weight, of industrial strength and accessible to a large community of engineers trained in object-oriented programming.

Conclusions

JACK Intelligent Agents is a multi-agent framework that extends the Java language. The current version supports the BDI model, and its modularity enables extensions and different models to be easily supported.

JACK is an industry-strength product, providing a framework that takes a solution founded in artificial intelligence research into practical use. Compared with its "predecessors", e.g., the PRS and dMARS systems mentioned above and other similar agent frameworks available in the academic world, JACK is not a "pure" AI system. Instead, it constitutes a successful marriage between the vision of agent research and the needs of software engineering, bringing the power of agent technology and enriching the host language, Java.

We are confident that JACK will provide benefits both to the software engineer developing distributed systems and to the academic researcher.

Acknowledgements

The authors would like to thank Prof. Ramamohanarao Kotagiri of the University of Melbourne for his valuable suggestions.

References

[1] M. E. Bratman, *Intention, Plans, and Practical Reasoning*, Harvard University Press, Cambridge, MA (USA), 1987.

[2] M. P. Georgeff and F. F. Ingrand, "Decision - Making in an embedded reasoning system", *Proceedings of the International Joint Conference on Artificial Intelligence*, Detroit, MI (USA), 1989.

[3] M. d'Inverno, D. Kinny, M. Luck, M. Wooldridge, "A Formal Specification of dMARS", *INTELLIGENT AGENTS IV: Agent Theories, Architectures, and Languages*, M. Singh, M. Wooldridge, and A. Rao (editors), LNAI 1365, Springer-Verlag, 1998.

[4] A. S. Rao and M. P. Georgeff, "An Abstract Architecture for Rational Agents", *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning (KR'92)*, C. Rich, W. Swartout and B. Nebel (editors), Morgan Kaufmann Publishers, 1992.

[5] M. Wooldridge and N. R. Jennings, "Intelligent Agents: Theory and Practice, *The Knowledge Engineering Review*, vol. 10, no 12, pp 115-152, 1995.

http://www.agent-software.com.au
(c) Copyright 1998, Agent Oriented Software Pty. Ltd.

AgentLink Research Skills Database

One of the goals of WorkPackage 2 in AgentLink is to establish a research skills database for AgentLink members. Once constructed, this database can be used for finding partners for collaboration within AgentLink, including future joint applications for the fifth frame program.

Your important contribution to this process is to fill out an electronic form, which can be found at

http://www.dsv.su.se/~mab/AL/Index.html.

In the near future, you will be informed about the progress in the process of constructing the database. If you have any questions about this process, or about this questionnaire, do not hesitate to ask Magnus Boman (mailto:mab@dsv.su.se).

ZEUS: A Toolkit for Building Distributed Multi-Agent Systems

Divine Ndumu, Jaron Collis, Gilbert Owusu, Matt Sullivan, and Lyndon Lee
Intelligent Systems Research, BT Laboratories, UK
ndumudt@info.bt.co.uk

ZEUS is a toolkit for building deliberative and goal-directed multi-agents systems for task-oriented domains such as service provisioning, resource/process management, and supply chain management. An evaluation copy of the toolkit will be available for public download from our web site from around mid-December 1998.

The toolkit allows users to:

- *configure* a number of different agents of varying functionality and behaviour;
- *organise* the agents in whatever manner using system-supplied organisational relationships;
- *imbue* each agent with selected system-supplied and/or user-defined communicative and co-ordination mechanisms;
- *supply* each agent with appropriate application-specific problem solving code; and
- automatically *generate* the agent executables.

In addition, the toolkit provides predefined information discovery agents such as nameserver and facilitator agents, as well as extensive facilities for visualising and debugging multi-agent societies.

Inside the ZEUS Toolkit

The ZEUS toolkit consists of a set of components, written in the Java programming language, that can be categorised into three functional groups as depicted in Figure 1: an agent component library, an agent building tool and a suite of utility agents comprising nameserver, facilitator and visualiser agents.

The Agent Component Library is a collection of classes that form the building blocks of individual agents. Together these classes implement the application-independent *agent-level* functionality required of collaborative agents. The contents of this library implement various aspects of agent functionality, such as communication, ontology, and social interaction. For communication the Agent Component Library provides:

- a performative-based agent communication language, ZEUS supports KQML and the FIPA ACL;
- an asynchronous socket-based message passing system;
- an editor for describing domain-specific ontologies ¾ the domain concepts that are defined using the ontology editor are used as part of the content language within the ACL; and

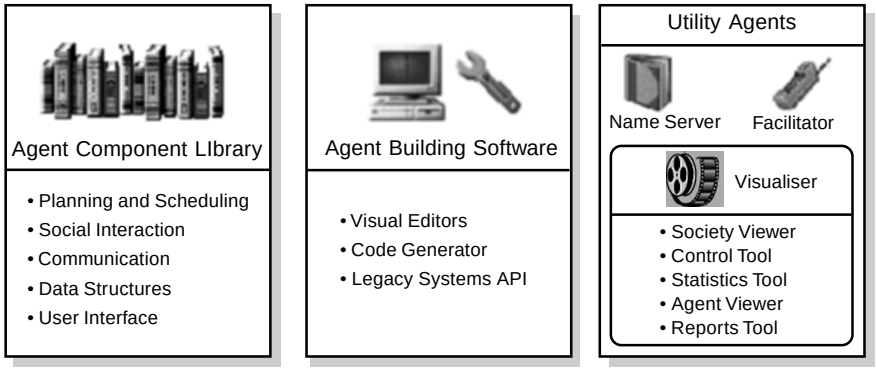

Figure 1: Components of the ZEUS agent building toolkit

- a frame-based knowledge representation language for representing domain concepts.

Next, for reasoning and multi-agent co-ordination, the Agent Component Library provides:

- a general purpose planning and scheduling system suitable for typical task-oriented application domains, and the co-operative problem-solving inherent to these applications, and
- a co-ordination engine that controls the social behaviour of an agent, i.e. when and how it interacts with other agents and the types of contracts it sets up with them.

The functioning of the planner and co-ordination engine are influenced by the agent's knowledge context, i.e. its available resources and competencies, its organisational relationships with other agents and its available co-operation strategies. So, to support these two components, the Agent Component Library also provides:

- a library of predefined re-usable co-ordination protocols, currently contract-net and various auction protocols.
- a number of predefined organisational relationships. The current set of relationships includes *superior*, *subordinate*, *co-worker* and *peer* relations.
- databases for storing the resources and competencies of an agent.

Together, the components of the Agent Component Library enable the construction of an application-independent generic ZEUS agent that can be customised for specific applications by imbuing it with problem-specific resources, competencies, information, organisational relationships and co-ordination protocols. Figure 2 shows the architecture of the generic ZEUS agent, which includes the following components:

- a Mailbox and Message Handler that handle communications between the agent and other agents.
- a Co-ordination Engine that makes decisions concerning the agent's goals,

e.g. how they should be pursued, when to abandon them, etc. It is also responsible for co-ordinating the agent's interactions with other agents using its known co-ordination protocols and strategies.

- an Acquaintance Database that describes the agent's relationships with other agents in the society, and its beliefs about the capabilities of those agents. The Co-ordination Engine uses information contained in this database when making collaborative arrangements with other agents.
- a Planner and Scheduler that plans the agent's tasks based on decisions taken by the Co-ordination Engine and the resources and task specifications available to the agent.
- a Resource Database that maintains a list of resources (referred to in this paper as *facts*) that are owned by and available to the agent. The Resource Database also supports a direct interface to external systems, which allows it to dynamically link to and utilise proprietary databases.
- an Ontology Database that stores the logical definition of each fact type ¾ its legal attributes, the range of legal values for each attribute, any constraints between attribute values, and any relationships between the attributes of the fact and other facts.
- a Task/Plan Database providing logical descriptions of planning operators (or tasks) known to the agent.
- an Execution Monitor that maintains the agent's internal clock, and starts, stops and monitors tasks that have been scheduled for execution or termination by the Planner/Scheduler. It also informs the Planner of successful and exceptional terminating conditions of the tasks it is monitoring. In order to manage tasks, the Execution Monitor also has a direct interface to external systems. It is assumed that the domain realisations of tasks are external programs.

Building Agents with ZEUS

With ZEUS, application-specific agents can be constructed by specialising the generic ZEUS agent. To facilitate the rapid development of multi-agent systems, the ZEUS toolkit provides a visual development environment that supports a high-level agent development approach. By using our approach, the multi-agent system developer does not need to know the details of the Agent Component Library in order to develop working agent systems.

At the highest level of abstraction, the ZEUS agent design approach requires developers to view an agent as composed of three layers: a *definition layer*, an *organisation layer* and a *co-ordination* layer. At the *definition layer*, the agent is viewed as an autonomous reasoning entity, i.e. in terms of its competencies, rationality model, resources, beliefs, preferences, etc. At the *organisation layer* it is viewed in terms of its relationships with other agents, e.g. what other agents it is aware of, and what abilities it knows they possess. At the *co-ordination* layer the agent is viewed as a social entity, i.e. in terms of its co-ordination and negotiation techniques. This agent model is supplemented with the protocols that implement inter-agent communication and an application programmer's interface that enables the agent to be linked to the external programs that provide it with resources and/or implement its competencies.

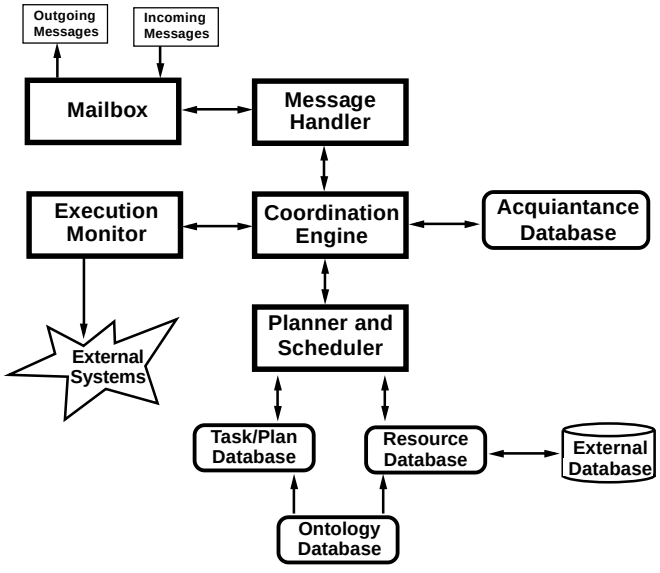

Figure 2: The architecture of the generic ZEUS agent

The stages of our agent creation approach are derived from this abstract high-level conceptualisation of what makes an agent. The stages are (i) domain study (which involves candidate agent identification and domain ontology specification), (ii) agent definition, (iii) task definition, (iv) agent co-ordination protocols specification, (iv) agent organisation, and (vi) domain-specific problem solving code production. Steps 1-5 are performed iteratively until the developer is satisfied that the final suite of agents accurately capture the details of the problem being modelled.

The ZEUS Agent Generator is a suite of integrated editors that support the ZEUS agent design approach. To facilitate ease of use, the editors have been designed to enable users to interactively create agents by visually specifying their attributes. The current suite of editors includes:

- An Ontology Editor for defining the ontology items in a domain. Concept categories - referred to as fact templates - can be created for application domains, with the concepts related to one another as appropriate through object-oriented style inheritance and/or composition. Fact objects are defined in terms of their attributes and the valid value ranges for each attribute.
- A Fact/Variable Editor for describing specific instances of facts and variables, using the templates created using the Ontology Editor.
- An Agent Definition Editor for describing agents logically. This involves specifying each agent's tasks, its initial resources, and the dimensions of its plan diary.
- A Task Description Editor for specifying the attributes of tasks and for graphically composing summary tasks.
- An Organisation Editor for defining the organisational relationships between

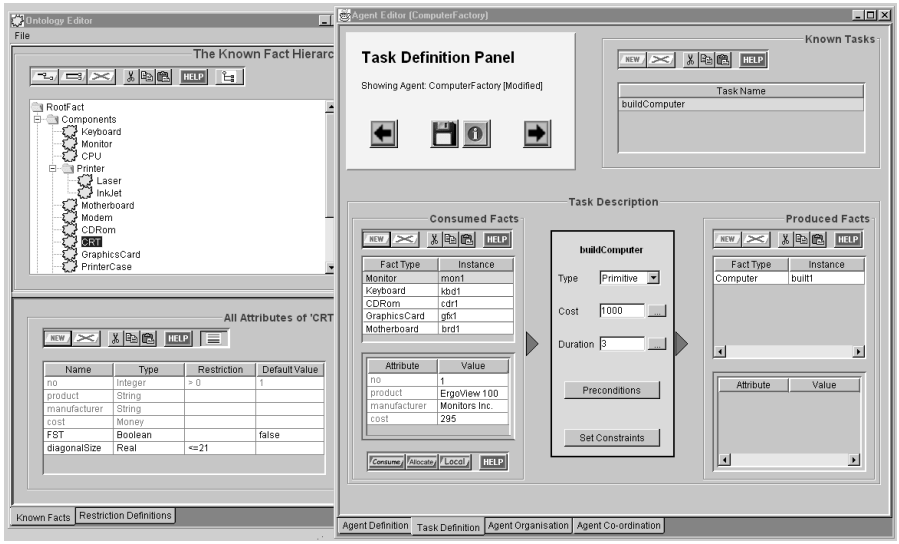

Figure 3: The ZEUS Agent Generation Environment. The window on the left hand side is the Ontology Editor, while that on the right hand side is the Task Definition Editor

agents, and agents' beliefs about the abilities of other agents.

- A Co-ordination Editor for selecting the set of co-ordination protocols with which each agent will be equipped.

Thus, in order to generate the code for a specific application, the ZEUS Code Generator tool *inherits* code from the Agent Component library, and *integrates* it with the data fromthe various visual editors. The resulting programs can be compiled and executed normally. Figure 3 is a screenshot of the ZEUS Agent Generation Environment.

The ZEUS Utility Agents

The ZEUS suite of utility agents consists of a *nameserver* and a *facilitator* agent for information discovery and a *visualiser* agent for visualising or debugging societies of ZEUS agents. A ZEUS agent society may contain any number of these utility agents, with at least one nameserver agent.

Nameserver agents are simplified ZEUS agents, as they only require the components that enable them to receive and respond to requests for the addresses of other agents. Another responsibility of Nameserver agents is to maintain a society-wide clock, which other agents use to synchronise their activities.

Facilitator agents store information about the abilities of other agents. This store is updated periodically by the facilitator asking each agent about its current abilities; alternatively individual agents can advertise their abilities to facilitators. So, when an agent wants to find other agents that have a particular competence, they can simply send an appropriate query message to a facilitator agent.

Visualiser agents can be used to view, analyse or debug societies of ZEUS agents. They function by querying other agents about their states and processes, and then collating and interpreting the replies to create an up-to-date model of the agents' collective behaviour. This model can be viewed from different perspectives through visualisation tools provided by the visualiser agents. The current tools include:

- a Society Viewer that shows all the agents in a society and their organisational inter-relationships. It can also show the messages exchanged between the agents during problem solving.
- a Reports Tool that shows the society-wide decomposition/distribution of active tasks and the execution states of the various tasks.
- an Agent Viewer that enables the internal states of agents to be observed and monitored.
- a Control Tool that is used to remotely review and/or modify the internal states of individual agents. Thus, an agent's behaviour can be redefined at runtime

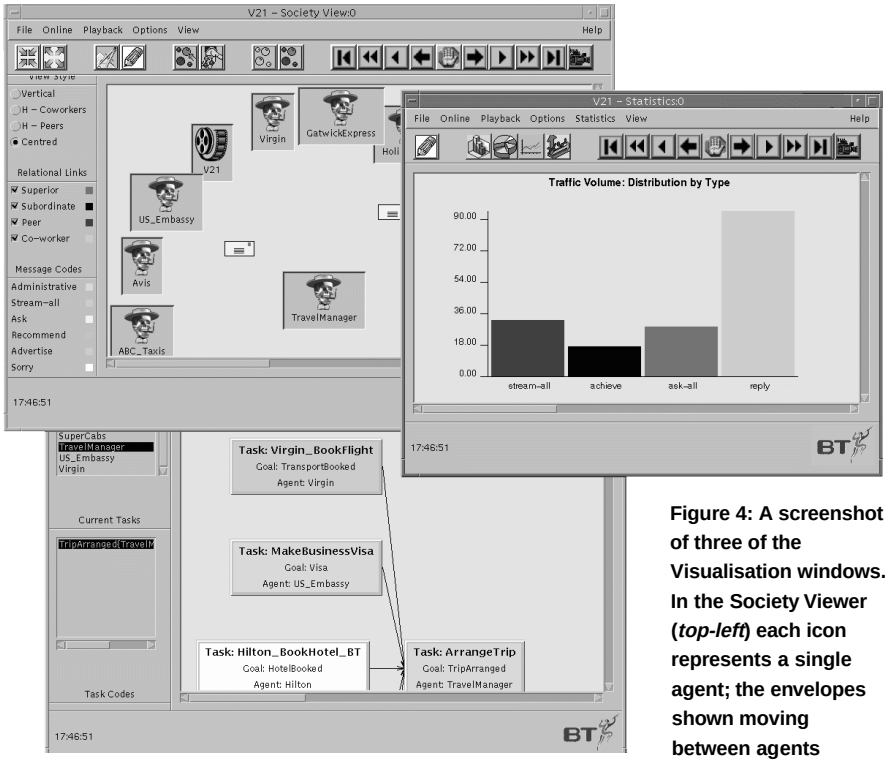

Figure 4: A screenshot of three of the Visualisation windows. In the Society Viewer (*top-left*) each icon represents a single agent; the envelopes shown moving between agents signify the transmission of messages. The toolbar buttons on the top left of the Society Viewer window control the position and visibility of the agent icons, while the buttons on the top right control the 'video replay' facilities. The Statistics Tool (*middle-right*) shows a histogram analysis of the type of messages being exchanged during the current session. The Reports Tool (*bottom*) shows the names and states of the sub-tasks that form the task currently being attempted by a sub-group of the agents

by using this tool to modify its task, resource, or organisational databases, in this regard, the control tool is effectively an online version of the Agent Building Software. This tool also facilitates administrative management of agent societies, e.g. agents can be killed or suspended, they can be given news goals, or their old goals can be modified.

- a Statistics Tool that displays individual agent and society-wide statistics in a variety of formats.

The multi-perspective visualisation approach provided by the visualisation tools gives users the flexibility to choose what is visualised, how it is visualised and when it is visualised. The visualisation tools are generally used *online*, to visualise the interactions in a multi-agent society live, as they happen. However, the society, report and statistics tools can also operate *off-line* by recording agents' interaction sessions to a database. Once stored, recorded sessions can be replayed, video-recorder style, using the forward and rewind buttons. Figure 4 is a screenshot of three of the visualisation tools.

Availability

From mid-December 1998, an evaluation copy of ZEUS will be available for public download from our web site:

http://www.labs.bt.com/projects/agents

The site also contains copies of ZEUS-related papers and descriptions of other agent-related projects by our group.

ACTS CLIMATE: An Overview

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

Today, the agent paradigm and emerging agent technologies are considered key for the implementation of flexible and scalable solutions for an open services market within the information society. The Cluster for Intelligent Mobile Agents for Telecommunication Environments - CLIMATE - represents a pool of Agent Technology related projects within the European Union collaborative research and development programme on Advanced Communications Technologies and Services - ACTS (http://www.infowin.org/ACTS). Most of these projects are located within Domain V focussing on Service Engineering, Security and Communications Management.

CLIMATE has been formed in Spring 1998 and currently comprises 14 core projects, which investigate until the end of this millennium the usage of Agent Technologies in various application areas, such as service control in fixed and mobile networks, telecommunications management, electronic commerce, multimedia applications, etc. The incorporation of emerging agent standards, particularly the OMG Mobile Agent System Interoperability Facility (MASIF) and the Foundation of Physical Intelligent Agents (FIPA) specifications, is key in this context. Furthermore the cooperation with related agent projects within ESPRIT and EURESCOM is considered very important in order to pave the way for the joint 5th Framework programme.

CLIMATE and its sub-clusters meets every three months in conjunction with the quarterly held ACTS Concertation Meetings (ACMs) in Brussels. The main work is performed via email lists.

CLIMATE Mission

CLIMATE's mission is to optimise the information exchange and to promote co-operation between these projects in order to enable the harmonisation of work, which ideally will result in a flexible common agent middleware, which could be used for different application domains. An important aspect of CLIMATE is the collaboration with other agent projects, particularly ESPRIT and EURESCOM projects, which are considered as associated CLIMATE projects. However, also other external projects may become associated members.

CLIMATE promotes the ACTS agent project activities and results towards the outside world (i.e., standard bodies, fora, industry, etc.) taking advantage of adopting an integrated view across all projects. It is envisaged that CLIMATE is taking an active part in contributing to relevant agent standards (e.g., OMG, FIPA) and telecommunication standards (e.g., IN, TMN, UMTS standardisation).

CLIMATE Structure

CLIMATE is structured into four subclusters in order to optimise exchange of information and collaboration, taking into account the diversity of projects in terms of agent technology used and application area addressed.

The mission of the IN & Mobility Subcluster is to explore the application of Intelligent Mobile Agent Technologies in the context of IN-based and mobile telecommunication environments. Focal points of research include extension of existing IN and mobile architectures, including mobile devices and smart cards by introduction of mobile agent platforms, in order to provide ubiquitously new applications, comprising video on demand, UMTS virtual home environment, customer profile management, mobility support, financial services, and dynamic provider selection.

The Communications Management subcluster is looking at the impact of mobile and intelligent agent technologies on the design of service and network management services. Besides the development of agent-based management platforms, the management areas addressed by the projects forming this subcluster include IN/SS7 load control, connection admission control for ATM networks, accounting and charging services for fixed and mobile environments, and service reservation.

The End-to-End Agent Systems subcluster is concerned with agent software designs which are concentrated in "end-systems" rather than within a network. The ACTS projects involved span a variety of application domains but

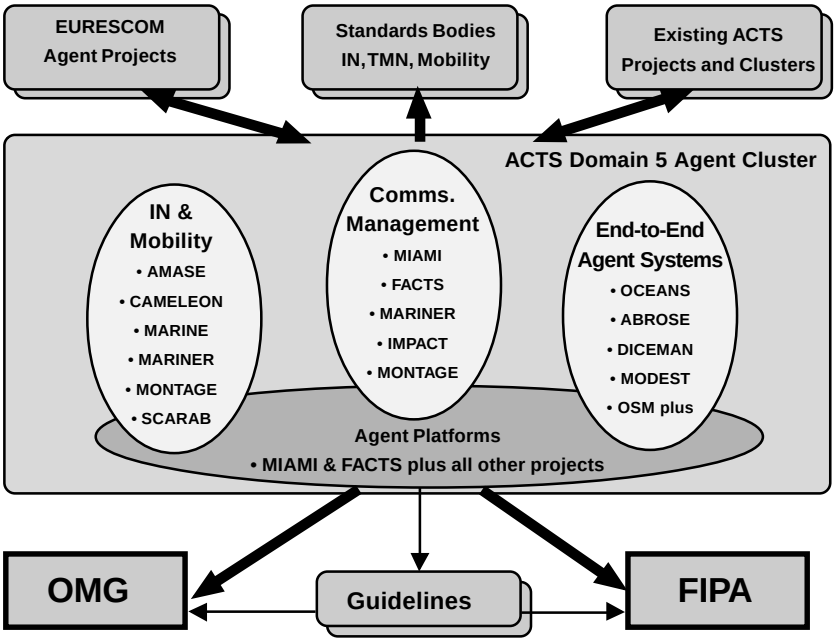

Figure 1: CLIMATE Structure

recognise common DAI issues which can be illuminated optimally in this inter-project forum. To this end, the sub-cluster seeks to analyse selected design topics in relation to agent negotiation, knowledge representation, and human-to-agent interaction. Considerations, will where possible, include relevant standards with a view to comment and input from member projects and/or the sub-cluster as a whole.

The Agent Platform subcluster is providing a forum of information exchange on agent platform issues, comprising experiences in the usage of available platform products, extension of basic platform capabilities as well as application related extensions.

CLIMATE Core Projects

Agent-based Brokerage Service Environment (ABROSE) aims to design, implement and integrate an agent based brokerage service relying on the usage of intelligent agents for the purpose of mediation, enabling User agent support for requests and navigation, Provider agent support for Offer registration and propagation. Through demonstrator sites (FT, Onyx) ABROSE wants to prove that Agent technology is improving the performance and the quality of mediation through usage of collaborative adaptive agents. WWW: http://b5www.berkom.de/ABROSE/

Agentified Mobile Access to Multimedia Services (AMASE) will investigate how to enhance an existing agent platform in order to support stationary and mobile agents for wireless mobile communication environment. The special focus of AMASE is on:

- High degree of user mobility between various networks, devices and physical location.
- Small and mobile devices for PC, personal digital assistants and GSM with SMS transmit and receive capabilities.

WWW: http://b5www.berkom.de/amase/

Communication Agents for Mobility Enhancements in a Logical Environment of Open Networks The CAMELEON project investigates the application of Agent Technologies for the purpose of providing service ubiquity. Mobile and Intelligent Agent Technologies will be combined to enable Service Roaming in heterogeneous networks. Three different types of services will be implemented. An Adaptive Profile Manager, a Virtual Address Book and a Flexible Financial Service. The applications will feature the Virtual Home Environment in UMTS based network environments. WWW: http://www.comnets.rwth-aachen.de/~cameleon/

FIPA Agent Communication Technologies and Services The goal of the FACTS project is to validate the work of FIPA and other standards groups by constructing a number of demonstrator systems based on its proposed standards. The focus of the project is in the interaction between differently-implemented agents. The project will be structured around two development cycles. During phase I, agent interoperability will be tested primarily within each of three application areas. These areas are: 1. audio-visual broadcasting and entertainment; 2. service reservation; and 3. electronic commerce (a travel-based example has been selected). During phase II, agent interoperability will be tested between the different application areas. WWW: http://www.labs.bt.com/profsoc/facts/

Multi Agent-based ATM Management IMPACT's overall objective is to implement control strategies on an ATM test bed as a society of interacting/co-operating agents, using the EXPERT platform in Basel on which realistic strategies can be implemented and tested. The application will concentrate on Connection Admission Control (CAC), introduce an accounting/charging agent and validate the FIPA model for agent management and communication. WWW: http://www.acts-impact.org/

Mobile Agent Environment for Intelligent Networks (MARINE) is an IN project aiming at distributing intelligence in the network by applying DOT and MAT to the network environment. In particular, the project considers MA-based deployment of IN service logics offering services to both fixed and mobile users. Existing IN devices are enhanced in order to adapt them to the distributed environment. A trial is foreseen in order to validate project choices. WWW: http://www.italtel.it/drsc/marine/marine.htm

Multi-Agent Architecture for Distributed-IN Load Control and Overload Protection The MARINER project will assess the usefulness of agent technology for the enhancement of the performance and management of complex distributed telecommunications network environments. The project will focus on the specification, development and trials of a multi-agent system for load control of

distributed, CORBA-based Intelligent Networks. WWW: http://www.teltec.dcu.ie/mariner

Mobile Intelligent Agents for Managing the Information Infrastructure The MIAMI projects context is the emerging Open European and Global Information Infrastructure (EII and GII) which is characterised by its increasing distribution, its dynamic nature, and the complexity of its resources. To manage such an environment, increased intelligence in management solutions and the mobility of such solutions are becoming major requirements. Within this context, the MIAMI project is going to focus on the creation of a unified mobile intelligent agent (MIA) framework by validating, refining and enhancing the OMG MASIF standards according to the requirements for an Open EII, to develop MIA based solutions for the management of the Open EII and for the provision of advanced communication and information services, to create a reference implementation of the Unified MIA Framework and solutions in order to evaluate the service solutions in a Pan-European business environment. WWW: http://www.fokus.gmd.de/ima/miami/

Agent-based Personal Mobility Management (MONTAGE) applies agent technology for service provisioning in an environment that involves mobile users and multiple competing service providers. Aspects of mobility to be covered include selection of appropriate service provider on the basis of service requirements and offerings, and ubiquitous service provision. Creation of agents for mobility support as well as agents for accounting and charging in a retailer federated context are envisaged by the project. WWW: http://montage.ccrle.nec.de

Open Communication Environment for Agent-based Networked Services The main OCEANS objective is to develop and exploit innovative software agent technologies in order to enhance interactive multimedia services delivered to residential users. With respect to conventional online services, agent-based applications are intended to exhibit capabilities such as autonomy, proactiveness and personalization. Special focus is given to terminal-side aspects, including interfaces and protocols for communicating with remote agents as well as agent-oriented user interface metaphors. WWW: http://www.sintef.no/ACTS-OCEANS

Open Service Model for Global Information Brokerage and Distribution The OSM project has build the next generation framework for globally distributed electronic commerce and virtual trading market systems. The project has developed a set of advanced Java based desktop components supporting catalogue browsing, service inspection, monitoring and construction tools. Below the desktop the project has developed a set of CORBA compliant facilities providing innovative techniques for federated service management. This suite of commerce facilities covering commercial service management, cataloguing, brokering and negotiation enabling globally open competitive electronic marketplaces.WWW: http://www.osm.net

Smart Card and Agent-enabled Reliable Access The SCARAB project focuses on the synergy between smart card and agent technology in an open, distributed and secure service architecture. It studies the use of smart cards for seamless access to a multitude of telecommunication services and intends to take benefit from the synergy between smart cards and agents for reducing the complexity and diversity of user interfaces as implemented on a multitude of telecommunication terminals. WWW: http://www.scarab.montrouge.tt.slb.com:65530

Distributed Internet Content Exchange with MPEG-7 and Agent Negotiations (DICEMAN): Content holders have vast quantities of material which they would like to sell. Content consumers have a growing need to access and purchase this content. However, today,s business model is primarily based on paper, video tapes and CDs, and the postal system, making it slow, expensive and cumbersome. DICEMAN intends to deploy existing and emerging technologies to provide a open, standards-based solution to this problem. Key technologies include: MPEG-7, FIPA and the Internet. WWW: http://www.teltec.dcu.ie/diceman

Multimedia Object Descriptors Extraction from Surveillance Tapes (MODEST) defines and develops a framework for analysing video sequences in order to obtain high-level semantic scene interpretation. Its technical approach is to segment, track and index, in a co-operative way, moving objects in video scenes. Results will be demonstrated on a video surveillance application involving a camera network. Due to the confluence position between MPEG-4 (coding and transmission), MPEG-7 (indexing) and FIPA's Intelligent Physical Agents (for the high-level interpretation), the MODEST project will take advantage of the standardisation works achieved and will contribute to them by their integration in a specific hierarchical image analysis framework. WWW: http://www.tele.ucl.ac.be/MODEST

In addition to these Core Projects CLIMATE is open for so called "Associated Projects". These projects are invited to participate in the CLIMATE subclusters in order to exchange information on experiences with the usage of agent technology and its application to telecommunications applications. Today projects from ESPRIT and EURESCOM are registered as associated projects. If you are personally engaged in an agent-related R&D project and interested in making your project an associated CLIMATE project, contact the CLIMATE contact persons.

CLIMATE Achievements

One mission of the CLIMATE projects is to gain experience in the usage of agent platforms and knowledge how to enhance these in order to be used for specific application areas. However, additional key areas include Agent-based Service Engineering, Management and Control inside Agent Platforms. This issue also addresses the potential Integration of Mobile Agent and Intelligent Agent technologies. The Agent Platform subcluster is supposed to address these aspects in more detail. As a means for stimulation of information exchange between the CLIMATE projects the establishment of a CLIMATE *Agent Catalogue* based on the input coming from the projects and subclusters will be established.

Contacts to FIPA and the new OMG Agent Special Interest Group have been established through key CLIMATE projects, such as MIAMI and FACTS.

In addition, a CLIMATE *Dissemination Catalogue* has been established in autumn 98. This should help projects to identify, which project is targeting which Telecom Fora (e.g. in regard to IN, TMN,

UMTS, EC, etc.) and thus open the door for collaboration.

CLIMATE is organising a one day workshop in Barcelona, Spain in April 26th 1999. The workshop is entitled CLIMATE Experiences in Usage of Agent Platforms and Design of Agent-Based Applications – *Is Agent Technology ready to Use in the Business?* The purpose of the Workshop is to provide the first experiences in the Usage of Agent Platforms and Design of Agent-Based Applications gained from the ACTS CLIMATE projects to both the projects and the industry. One important question to be answered by this workshop is whether Agent Technology is ready to use in the field and performs better than traditional technologies. The workshop is held prior to the ACTS IS&N ´99 Conference, which features at the first day (April 27th 1999) a showcase dedicated to agent technologies and three sessions dedicated to agent related themes (still to be determined).

CLIMATE Contacts

Dr. Thomas Magedanz (CLIMATE Chairman)
IKV++ GmbH, Kurfürstendamm 173-174,
D-10707 Berlin, Germany
Email: magedanz@ikv.de

Alessandro Barbagli
(European Commission Contact Point)
DGXIII/B - ACTS Programme, Domain 5: IS&N,
Av. de Beaulieu 9, B-1160 Brussels
Email:
Alessandro.Barbagli@bxl.dg13.cec.be

CLIMATE Web site:
http://www.fokus.gmd.de/cc/ima/climate/

Project Report

ADE: An Architecture type-based Development Environment for Agent Application Systems

Mario Kupries
Hasso-Plattner-Institute for Software System Engineering
University of Potsdam, Germany
mkupries@acm.org

There are many agent-based application systems that are designed to gain information, such as stock exchange information systems, internet ordering systems and systems within electronic commerce, such as credit advisory assistants and virtual marketplaces. All such systems are distributed, heterogeneous, dynamically expandable and open.

Under the supervision of Prof. Erika Horn the Software Engineering working team at the Department of Computer Science, University of Potsdam has been doing research into the scientific-engineering substantiation and development of such complex agent application systems. This article introduces the team's research in this area by examining a project that deals with the mapping of mental models of agent application systems onto agent platforms. This mapping was identified as a crucial area for development during the AgentLink Software Engineering Special Interest Group kick-off meeting in Brussels, September 1998.

The approach

Software agents and agent systems are modelled object-orientedly using the latest findings of software architecture. This approach (see figure 1) describes a software architecture as a system of components whose interactions are realised via connectors [1]. In the architecture type-based approach the structural aspects of agent systems, classes of software agents, agent properties and agent connectors, are described together with operations for the composition, analysis and validation of these systems [2].

For classes of agent application systems special architecture types are derived by means of object-oriented techniques, and thus a hierarchical order of architecture types is evolved. This enables the systematic analysis, classification, substantiation and development of agent application systems and leads on to the prefabrication of special and re-usable building block and property classes for agent application systems. Instances of these classes are, for example, security agents, management agents, negotiation patterns, instruction connectors, itineraries, agendas, etc. [3].

Especially beneficial in the architecture type-based development of agent application systems is the explicit modelling of interactions between agents: the modelling of agent connectors. Agent connectors encapsulate in their protocols, among other things, types of speech acts. For the individual control of the agents' co-operation behaviour, such as a unilateral instruction, special connectors can be used. Depending on the respective behaviour and negotiation pattern, specialised connectors can be used for agent interactions [4].

Furthermore, the explicit modelling of property classes is of great advantage since classes of properties for agent

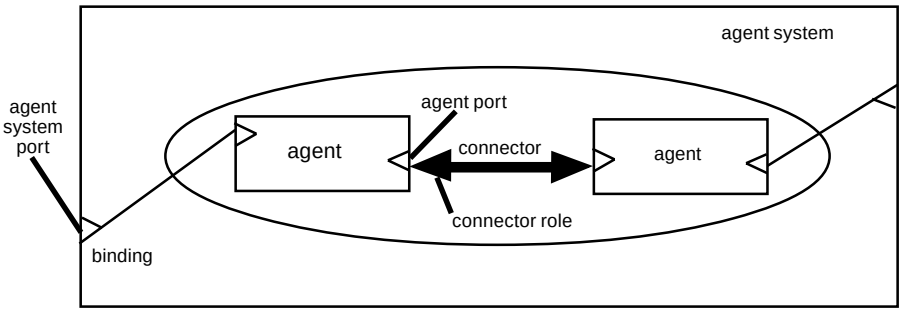

Figure 1: Architecture type 'Agent System'

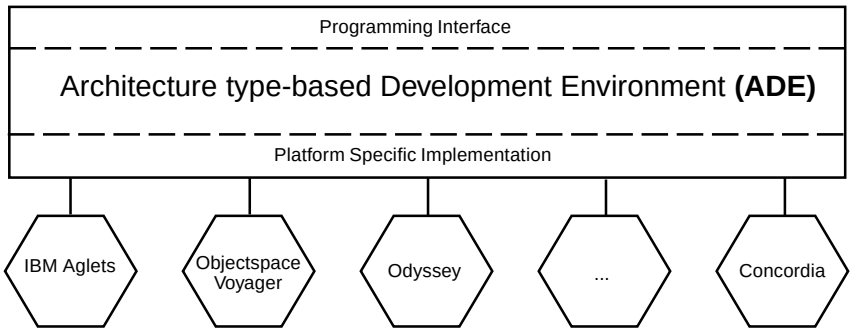

Figure 2: Principle of the architecture type-based agent development environment.

application systems can be modelled abstracted from the field of application.

The project

With the architecture type-based methodology, theoretical fundamentals for the implementation of computer-independent platforms for agent programming have been found. Existing platforms, such as IBM Aglets or Objectspace Voyager, do not support an architecture type-based approach and environment when developing agent application systems. As a result these platforms miss, amongst other things, the explicit modelling of agent interactions, or allow only a very limited number of property classes. The mapping of object-oriented hierarchies and models onto these platforms has reached very quickly its many limits and restrictions. The advantages of a substantiated methodology of an architecture type-based approach (such as the differentiation between building block and property classes, and the explicit modelling of agent interactions and the hierarchical prefabrication) are lost while transforming them onto existing platforms.

Because there are a lot of agent platforms, the development of another platform has not been attempted. To transfer existing concepts and technologies into products we have decided to encapsulate the existing agent platforms. To map the mental models and systems onto agent platforms, an architecture type-based agent development environment for the modelling, analysis and construction of agent application systems for various agent platforms called ADE – *"Architecture type-based Development Environment"* is being developed (see figure 2).

The ADE system

Figure 2 illustrates the strategy for integrating architecture type-based models of agent systems into existing agent platforms. The ADE corresponds to a *mediator* which consists of different layers (see figure 3). From a software engineering point of view, the mediator principle is about encapsulation and adaptation to a basis machine.

The elements of *the architecture type layer* are building block and property classes for the architecture type 'agent system' whose set is open. In this layer classes of agents and connectors are embedded together with their property classes. This layer is the interface between the mediator and the agent platforms. The *prefabrication layer* contains special, re-usable classes of building blocks and properties for special classes of agent application systems. These two layers enable the architecture type-based modelling of agent application systems - the modelling result is mapped onto agent platforms. To this end the ADE has got a functionality for mapping classes, converting models, enforcing consistency, evolution, ontology, navigation, organisation, etc. in the *transformation layer*. The ADE has a *database interface* to create and access persistent modelling results. Finally, The *interaction layer* is used for the human agent interaction. This layer is designed for assisting humans, store preferences and ontology of the users. It is ADE's interface to users.

The specification of the functionalities of the layers is done taking into consideration application-independent services which partly are the content of standards (e.g. Agent Name Service, Ontology Service). The ADE is open concerning the classes of agent application systems, the manufacturer and the platform type.

The ADE enables the creation of a *methodology* for the platform-independent and architecture type-based construction of agent-based systems with substantiated scientific-engineering methods. The ADE is a starting point for specialised construction technologies in agent applications.

The mediator ADE has been specified in the implementation-independent language CORBA-IDL. It can be realised platform-independently, e.g. using Java or C++. This way various realisation forms can be evaluated regarding the performance and efficiency of the transformation of architecture types on agent platforms.

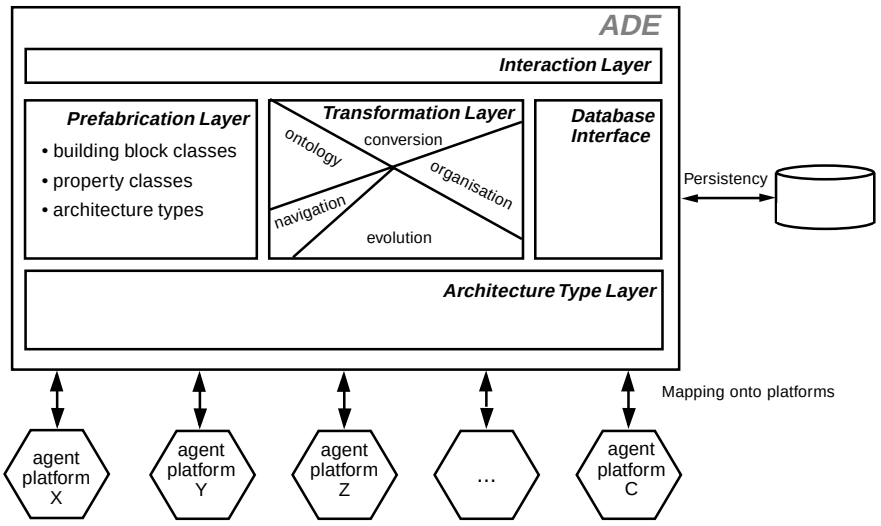

Figure 3: Internal structure of ADE

Special activities within ADE

The Software Engineering working team at the University of Potsdam has analysed special properties of agent technology and developed them further in terms of the architecture type-based use in agent application systems on the basis of ADE.

The work on the *co-ordination* of the co-operative behaviour of software agents deals with the development and prefabrication of special connectors. These agent connectors encapsulate protocols which control the agents' behaviour. Here the semantics of speech acts are mapped onto the behaviour patterns and action scopes of software agents. For agent *management* a decentralised approach is used providing each agent with its own communication and computation model.

Using typical examples from the fields of electronic commerce and banking, *security concepts* in agent systems are tested. At the same time concepts of reliable mobility are analysed. The prefabrication of special property classes of secure systems, such as authenticity and access control lists, is another focus of activity. *Mobility strategies* and the analysis of interoperable systems will be the content of future projects.

The use of architecture theory enables the *prefabrication* and re-use of the results of the scientific papers and their analysis as frameworks and patterns for agents and agent systems and the derivation of classes of agents, connectors and agent systems in the prefabrication layer [5].

Technology transfer

The architecture type-based approach proved to be successful when developing a system for credit advising. Furthermore, ADE will be used to develop large agent application systems, such as "TravelBot" - an agent-based system for travelling assistance. By re-using prefabricated models and ADE rapid prototyping can be achieved. ADE will be a registered trademark of the University of Potsdam.

Further information about the papers and results of the Software Engineering working team can be found on the WWW at http://www.cs.uni-potsdam.de/soft. Any comments are welcome!

References

[1] Shaw, M., Garlan, D. (1996) *Software Architecture. Perspectives on an Emerging Discipline.* Upper Saddle River: Prentice Hall.

[2] Horn, E., Schubert, W. (1993) *Objektorientierte Software-Konstruktion.* Hanser-Verlag, Wien, München.

[3] Horn, E., Kupries, M., Reinke, T. (1998) *Object-oriented software architecture types for the substantiation, development and prefabrication of agent application systems.* 11th International Conference on Software Engineering & its Applications.

[4] Kupries, M., Noseleit, C. (1998). *Co-ordinating Software Agents using Speech Acts.* University of Potsdam, Preprint.

[5] Horn, E., Kupries, M., and Reinke, T. (1998) *Properties and Models of Software Agents and Prefabrication for Agent Application Systems.* Software Technology Track of the Thirty-Second Hawaii International Conference on System Sciences.

The Mole Mobile Agents Research Group

Markus Schwehm
University of Stuttgart – IPVR, Germany
schwehms@informatik.uni-stuttgart.de

The Mole research group was founded early 1995 in order to do research on mobile agents. In Summer 1996 the first public release of our Mole mobile agent system was published on the web. This was the first mobile agent system written completely in Java. After providing the basic functionalities needed for agent mobility, further development of the Mole mobile agent system has focused on infrastructure and concepts necessary to support electronic commerce. Today the research group consists of Joachim Baumann, Fritz Hohl, Markus Strasser (research assistants), Markus Schwehm (project leader) and some loosely associated researchers who use - rather than develop – the system. Since 1996, 29 student projects and master theses have been completed. The group has published about 20 papers. Since September 1998 the third release of the Mole mobile agent system can be downloaded from our web site.

The new release now incorporates advanced concepts like agent termination and communication sessions. Furthermore the Mole research group has presented the mobile agent system on the CeBIT 1997, has helped organizing the First International Workshop on Mobile Agents '97 in Berlin and has hosted the Second International Workshop on Mobile Agents '98 in Stuttgart (please see the corresponding report in this issue). Concerning the application of agents in the context of electronic commerce, three main branches of research have been investigated:

Transaction based agents

(Markus Strasser) A prerequisite for the use of mobile agents in an electronic commerce setting is a reliable execution of agents independent of communication and node failures of the underlying computer network. We have proposed and implemented a fault-tolerant protocol that ensures the exactly-once execution of an agent. The protocol in particular guarantees that the agent cannot be trapped on a crashed node until the node is rebooted but will resume execution on alternative nodes after a timeout. This is achieved by monitoring the agent using observers, which will select a new node and reactivate the agent execution once the original agent becomes inactive. The exactly-once execution of agents is guaranteed by executing the agent within a transaction and by embedding a voting

protocol into the two phase commit protocol of the transaction. This research was done within a cooperation with Tandem Computers Inc., who also provided a reliable, transaction oriented hardware platform (Tandem Himalaya) for the implementation of the agent system.

Security Issues in mobile Agents

(Fritz Hohl) Security is a crucial issue if agents are to be applied in the context of electronic commerce. While many security issues can be addressed using techniques from distributed systems, the application of mobile agents raises new issues that have no counterpart in distributed systems. In particular for the protection of mobile agents against malicious hosts new security mechanisms have to be developed. While the problem is frequently ignored in practice ("Don't take your valuables into insecure terrain"), a formally justified security mechanism seems to be out of reach for practical applications. Our research focuses on a practicable approach that protects the agent for a limited time, so that at least one transaction can be completed securely. The approach is based on the obfuscation of code so that a potential attacker cannot manipulate the code in a meaningful way, in conjunction with restricting the lifetime of the agent's code and data. This research project has been funded by the Deutsche Forschungsgemeinschaft (dfg).

Orphan detection and agent termination

(Joachim Baumann) If agents are to be used in a commercial setting, their user might eventually have to pay for the resources used by an agent. It is therefore in the interest of both the agent owner and the agent platform provider to detect orphan agents whose resource consumption cannot be charged to anyone or conversely to actively terminate agents in order to prevent the consumption of expensive resources. Our research combines two basic approaches for orphan detection and agent termination, namely the energy approach and the path approach into one powerful approach called shadow protocol. The shadow protocol uses the idea of a placeholder (shadow), which is assigned by the agent system to each new agent. While the chain of shadows and shadow proxies work similar to the path approach, the interaction between the shadow and the corresponding agent uses the energy approach. This project has been partially funded by the Swiss Research Foundation.

For more information and downloading of the Mole mobile agent system see http://www.informatik.uni-stuttgart.de/ipvr/vs/projekte/mole.html

The DAI Unit at Queen Mary and Westfield College

Nick Jennings and Mike Wooldridge
Queen Mary and Westfield College, London, UK
{N.R.Jennings, M.J.Wooldridge}@elec.qmw.ac.uk

The Distributed AI Unit at Queen Mary & Westfield College (University of London) is one of Europe's largest research groups in the area of intelligent agents and multi-agent systems. It is headed by Nick Jennings and Mike Wooldridge and contains some 15 members.

The DAI Unit was formed in 1988 with the award of ARCHON, an ESPRIT II project. ARCHON was one of ESPRIT's largest ever projects (over 100 person years of effort) and it was tasked with developing a number of real-world applications using multi-agent system technology. ARCHON was successful in this endeavour, developing an agent architecture for industrial process control applications and seeing applications fielded in the domain of electricity management. From this time, the group has had a strong applied research emphasis, having developed agent systems for a range of industrial and commercial domains. In particular, systems have been developed for business process management, telecommunications network management, process control, patient care, concurrent engineering, scientific data interpretation, digital libraries and electronic commerce. These research projects have typically been undertaken with industrial collaborators including: Atlas Electronic, BT, BHP, CERN, Iberdrola, ICI Engineering, the Imperial Cancer Research Fund, Marconi Communications Systems, and Nortel.

The DAI Unit also has a strong tradition of research in the foundations of multi-agent systems. It was one of the pioneers of using explicit models of joint intentions to guide cooperative problem solving activity, and in addition has strengths in the use of non-classical logics for modelling multi-agent activity. Contributions have also been made to the theories of coordinated problem solving, social reasoning, and negotiation and argumentation. Recently, the group has become more involved in issues related to agent-oriented software engineering: this being used as the route to downstream the basic research activities into applied systems.

The group currently participates in a number of European and nationally funded projects, including:

Trilogy: Using agents in a virtual laboratory (UK EPSRC funded) This collaborative project is concerned with the design and implementation of a multi-agent system for managing the information and services (e.g. use of software tools) within a virtual research laboratory across the sites of three UK universities (QMW, University of Durham, and University of Strathclyde).

CAC: Market Based Control for the Management of Telecommunication Networks (UK EPSRC funded) This project investigates a novel approach for managing Connection Admission Control (CAC) in Asynchronous Transfer Mode (ATM) telecommunication networks. The project addresses the application of an intelligent multi-agent system, in which agents are modelled as self interested decision makers in a computational market, to network level resource allocation.

Practical Negotiation for Electronic Commerce (UK EPSRC funded) This project aims to develop techniques for negotiation in second-generation electronic commerce systems. The project will adapt techniques from game and economic theory, using in particular the techniques of mechanism design.

Proof Methods for Temporal Logics of Knowledge (UK EPSRC funded) Temporal logics of knowledge and belief have been widely used to reason about multi-agent systems since the mid 1980s. Despite this level of activity, comparatively little effort has been devoted to developing proof methods for such logics. This project aims to develop a range of proof methods for temporal logics of knowledge, including tableau, resolution, and labelled deductive systems.

IMPACT: Implementation of Agents for CAC on an ATM Testbed (EC ACTS funded) The overall objective of IMPACT is to implement control strategies on an ATM testbed as a society of interacting/co-operating agents. The work will use the EXPERT platform in Basel and will interconnect the switch on which the agents are implemented with the existing testbed network in order that the performance of the agent approach may be trialled.

Negotiating Agents for Telecommunications Management (Nortel Technology funded) The project deals with Service Level Agreement negotiation in telecommunications systems.

Brokering in an Information Economy (Marconi Communications Systems funded) The project will investigate the roles of brokering agents in an on-line trading environment.

Agent-Based Software Engineering Investigates the issues involved in using agents and multi-agent systems as a software engineering paradigm for building complex, real-world systems.

The DAI Unit has always had a strong presence in the professional aspects of the multi-agent system field. It is the coordinating site or the AgentLink Network of Excellence and it was responsible for initiating the International Journal of Autonomous Agents and Multi-Agent Systems. Group members have started or played prominent roles in the following international conferences and workshops: the International Conference on Multi-Agent Systems (ICMAS), the Practical Application of Autonomous Agents and Multi-Agent Systems (PAAM) conference, the Autonomous Agents conference, the Agent Theories, Architectures and Languages (ATAL) workshop series, and the UK Special Interest Group on Multi-Agent Systems (UKMAS - formerly FOMAS). Since 1988, members of the DAI group have published nine books on various aspects of agent technology.

You can find out more about the DAI unit from the group's WWW site: http://www.elec.qmw.ac.uk/dai/

Conference Report

The Second International Workshop on Cooperative Information Agents (CIA-98)

Matthias Klusch
Technical University of Chemnitz, Germany
klusch@informatik.tu-chemnitz.de

Research and development of intelligent information agents can be seen as one of the key technologies for the Internet and the world-wide Web. The interdisciplinary CIA workshop series covers the whole thematic range of intelligent and collaborating information agents. It emphasises on the synergies between related research disciplines such as Advanced Database and Knowledge-based Systems, AI, Multi-Agent Systems, Distributed Information Retrieval, and Human-Agent Interaction. Each workshop in the CIA series focus on a few selected themes of particular relevance and actuality for R&D in information agents.

The second international workshop on cooperative information agents (CIA-98) was held from July 4 - 7, 1998, in conjunction with the Agents' World event in Paris - La Vilette, Cite des Sciences. Main themes of CIA-98 were adaptation, mobility and electronic commerce for information discovery on the Internet. The workshop had 100 registered participants (from more than 500 attendees of Agents' World) and 54 submissions. It was co-sponsored by Daimler-Benz AG (R&T Berlin, Germany) and George Mason University (Fairfax, USA), and supported by the Special Interest Groups on DAI and Database Systems of the German Computer Society (GI).

CIA-98 did feature 10 invited research and industrial lectures from experts in the field, and 2 panel discussions. Invited speakers were Sharma Chakravarthy (University of Florida, USA), Edmund Durfee (University of Michigan, USA), Joerg Mueller (John Wiley & Sons, UK), Marian Nodine (MCC, USA), Van Parunak (ITI CEC, USA), Sandip Sen (University of Tulsa, USA), Katia Sycara (Carnegie Mellon University, USA), Robert Tolksdorf (TU Berlin, Germany), Munindar Singh (North Carolina State University, USA) and Mike Huhns (University of South Carolina, USA). The 14 contributed regular talks were scheduled in 5 thematic sessions.

The opening session of the workshop was devoted to demonstrations of new systems of collaborating information agents in different application areas such as logistics, hypermedia information discovery and traffic information in urban settings. Besides, recent advances in the cooperative information system InfoSleuth and the UMDL service market agent society were reported. There was a broad consensus among the participants

of the workshop that information agent technology has already achieved a widespread acceptance as one of the key technologies for building modern global enterprise-wide information systems. This technology benefits in part from the progress made in advanced database and information system technology.

Electronic commerce (EC) on the Internet is steadily growing. The use of simple ShopBots and virtual bookstores on the Web became very popular in the past five years. But a more sophisticated agent-based trading remains to be a key challenge for economists, computer scientists and business managers as well. Systems of economically rational information agents might reshape the way we think about economic systems and business processes in an increasingly networked world.

In a special session on information agents and EC a variety of topics and issues of rational collaboration were addressed. It was emphasised that it is important to enhance mobile agents with EC capabilities such as trust and virtual bank services. Besides, although grounded techniques for utilitarian coalition formation and automated negotiation among self-interested agents are known (recent progress in this direction was reported by Tuomas Sandholm in a plenary talk at Agents' World), no fielded agent system for EC using these techniques is available.

A thoughtful analysis and simulation experiment on the dynamics of an information filtering economy was shown by a team from IBM Watson Research Center. One main result was that efficient self-organisation of broker agents (selling information to customers) into specialised "information service" niches can occur when communication and processing costs are balanced, but price wars may even occur.

During a panel discussion on rational information agents (panelists were Robert Guttman from MIT, Pablo Noriega from National AI Lab in Spain, Joerg Mueller and Onn Shehory from CMU Pittsburgh) it was particularly emphasised that there are already some appropriate standards for secure electronic transactions and profiling in use or on its way. These are in part used for agent-based trading in retail electronic commerce. Some hard problems remain to be solved, e.g., the gradual achievement of trustworthiness, and interoperability among heterogeneous and proprietary agent execution platforms.

Up to now only a few systems of collaborating adaptive information agents exist. Most of the talks concerning such kind of agents presented novel variations of collaborative and content-based information filtering for recommendation of relevant Web sites. Some other techniques for information discovery such as the application of Kohonen's self-organising neural network map, and domain experts using mobile agents for knowledge acquisition were proposed and strongly discussed.

In the ongoing RETSINA project at Carnegie Mellon University different levels of adaptive behaviour such as organisational and individual agent level are considered. Some encouraging experiments with agent cloning for adaptivity to performance and user profiling are done in that project.

Particularly interesting was the report on a bunch of experiments (at the University of Tulsa) on learning approaches by which agents can adapt to select information sources. The selection criteria in these ongoing experiments based on performance in terms of time and provided information quality. Probability-based load balancing in distributed systems appears to be quite close, but has a few different underlying assumptions. However, research in collaborating adaptive information agents still is a relatively uncharted territory.

Another issue of adaptivity has been attracted much interest to the CIA-98 participants: Socially intelligent agents. The vision is that such agents are capable to adapt themselves to cross cultural challenges such as different social norms, negotiation strategies and forms of human-agent interaction. It was reported that projects in the Multimedia Super Corridor initiative in Malaysia have started to tackle some of those challenges.

The talks in the last session of this workshop addressed recent advances in developing systems of mobile information agents. Although there is an ongoing discussion about what characterises a mobile and intelligent agent, how mobility can be achieved on heterogeneous platforms, and what the payoff from applying mobile agent technology in different application domains will be - it is certainly no science fiction anymore. Though, there are not many mobile *information agent* systems known. Therefore, the report on the design and current status of implementation of the ongoing MIAOW (Mobile information agents on the Web)/InfoSphere project of Siemens AG in Germany triggered much interest and discussion. The same with a first security proposal for both, server and agents in the IMPACT agent system architecture by the University of Maryland. Constraints on code, data and information access may be specified in a semi-decidable DATALOG-like language including functions for verification and consistency checking.

Moreover, the use of a special kind of software pattern templates including dependency information was proposed for the design of coordination among mobile agents. Although this is certainly a valuable idea of a kind of reusing and modelling coordination patterns, it has to be shown in near future that more complex coordination and collaboration mechanisms such as auctions, voting, temporary team formation and others may be sufficiently described by these structures.

In the closing panel discussion on adaptive and mobile information agents (panelists were Henry Lieberman from MIT, Robert Tolksdorf, and Munindar Singh) it was in particular pointed out that in open networks, the security problems and costs of potential solutions might outweigh the benefits of mobility. The question of security goes in both directions: How can database servers be protected from malicious actions of mobile agents, and, in turn, how can an information agent, packed with private data and information, be protected from servers and other agents while travelling through the Cyberspace? Despite some technical solutions exist it was common consensus among the panelists that there is still no satisfactory solution known.

Regarding the trend in personal assistants for the Web there is no doubt that the recent gain of popularity will remain and probably increase in the next years. A large amount of several kinds of adaptive user interface agents for the Web have been developed in the last

years, and more will come. But more important is that they are expected to be broadly used - the need of such agents seems to be pressing yet. One question to the panelists was: How might a personal assistant look like in the future? No common agreement. Advanced forms of user's guidance are important and might be realised by, e.g., techniques from Virtual Reality, synthetic characters, like animated creatures or believable avatars. In addition, the development of highly interactive interfaces may include automated speech and motion recognition such as used for affective computing. Another question is how this trend in single personal assistants will affect the R&D on systems of collaborating adaptive agents. What is the future of recommendation and reputation systems with adaptive profiling techniques? Just a niche in information agent technology?

In summary, the CIA-98 workshop was quite a success regarding the high-quality contributions, outstanding invited talks, large interest and viable discussions.

The proceedings of the CIA-97 and CIA-98 are published as volume 1202 and 1435, respectively, in the series of Lecture Notes in Artificial Intelligence from Springer Verlag. The CIA-98 workshop was organised by Matthias Klusch (TU Chemnitz) as General Chair, and Gerhard Weiss (TU Munich, Germany) and Larry Kerschberg (George Mason University, USA) as Co-Chairs.

The third CIA workshop (CIA-99) will be held in Uppsala nearby Stockholm (Sweden), July 31 - August 2, 1999, in conjunction with the IJCAI-99 conference. More information on the CIA-99 workshop may be found in the Web at http://www.informatik.tu-chemnitz.de/~klusch/cia99.html

Second International Workshop on Mobile Agents (MA'98)

Fritz Hohl and Markus Schwehm
University of Stuttgart – IPVR, Germany
Fritz.Hohl@informatik.uni-stuttgart.de

In September 9-11, 1998 the Second International Workshop on Mobile Agents (MA'98) was held on the Campus of the University of Stuttgart, Germany. The workshop was organised by the University of Stuttgart and supported by the Gesellschaft für Informatik (GI), the IEEE Computer Society, the Informationstechnische Gesellschaft (ITG) and the International Federation for Information Processing (IFIP). The Workshop was chaired by Prof. Kurt Rothermel from the Institute of Parallel and Distributed High-Performance Systems (IPVR) of the Faculty of Informatics. The goal of this workshop on Mobile Agents was to provide a forum for the presentation and discussion of the most recent research results in the field of Mobile Agents.

For the seven sequential sessions 21 submissions from ten countries (Germany, France, United Kingdom, Italy, Norway, Portugal, Switzerland, Israel, Japan and USA) had been accepted. Similar to the last Workshop on Mobile Agents held 1997 in Berlin more than one hundred people from all over the world participated in the workshop and visited the seven sessions about such diverse topics as *Mechanisms for Mobile Agent Systems*, *Mobile Agent Architectures*, *Mobile Agent Systems*, *Security*, *Communications* and two sessions about *Applications*. Each day was opened by an invited talk. The first invited talk was presented by Danny Lange from General Magic, Inc. about "Present and Future Trends of Mobile Agent Technology"; he covered topics from the status quo of mobile agent technology over standardisation efforts to future requirements. The second invited talk was given by Dag Johansen from the University of Tromso, Norway, about "Mobile Agent Applicability". Professor Johansen described an application for storm forecasts in Norway which utilises many advantages of mobile agent technology and runs profitable for many years now. The third invited talk was given by Luca Cardelli from Microsoft Research. He discussed "Wide-Area Languages", i.e. languages that support code mobility and also presented a formal model called "Mobile Ambients" that does not only model mobile agents but also other approaches to mobile computing. The high quality of the invited speakers was well received by the audience and initiated intensive discussions. The discussions did not only emphasise the large potential of mobile agents for future network applications e.g. in the field of information retrieval or electronic commerce, they also illuminated the technical problems that have to be solved before a commercial usage of mobile agent technology is feasible. Here in particular fault tolerance, security of agents, users and providers, and the lack of a widely installed infrastructure have to be mentioned.

The first session dealt with talks about orphan detection, fault tolerance and transparent migration of Java programs in the context of system mechanisms for mobile agents. The second session illuminated issues of system architectures with talks about requirements, design, standardisation and state capture of mobile computations. The discussion after the standardisation talk lead to an additionally scheduled panel session on standardisation on the next morning. The second day started with a first session about applications. The applications came from areas like traffic management, mobile communication networks, and mobile devices. The next session was devoted to existing mobile agent systems like the "Mobile Object Workbench", the "AgentSpace" system and the "muCode" system. As usual the session on security was discussed controversely with talks about intellectual property protection, integrity of agent based computations and the protection of computations of mobile agents. The sessions on communications dealt with agent-user communication, with negotiation protocols for mobile agents and the usage of tuple spaces. The final session again described applications for mobile agents for the management of telecommunication networks, for scalable service deploy-

ment and for the management of video conferences using active networks.

The proceedings for the Second International Workshop on Mobile Agents, edited by K. Rothermel and F. Hohl have been published by Springer-Verlag, Germany, as Lecture Notes in Computer Science 1477, ISBN 3-540-64959-X. Further Information about the Workshop can be obtained on the Web from

http://www.informatik.uni-stuttgart.de/ipvr/vs/ws/ma98/ma98.html

The next Workshop on Mobile Agents will take place September 26-29, 1999 in Monterey, California and will emphasise mobile agent applications. Further information about MA'99 can be obtained from

http://www.genmagic.com/asap/

What's Happening in AgentLink?

Mike Wooldridge
AgentLink, Queen Mary and Westfield College
M.J.Wooldridge@qmw.ac.uk

In this column, we aim to give you a brief summary of what else is happening in AgentLink. Since all this information is available in detail on the AgentLink WWW site we will only summarise the crucial points.

AgentLink Summer School '99

The centrepiece of AgentLink's teaching and training activities in 1999 will be a summer school. This summer school — the first European Agent Systems Summer School (EASSS'99) will be held in Utrecht, in the Netherlands on July 26-30, 1999. Utrecht is a beautiful historic city close to Amsterdam (40 minutes by train from Amsterdam airport, 30 minutes by train from Amsterdam centre). EASSS'99 will take place the week before IJCAI'99. Courses to be delivered at the summer school will cover the range of agent systems activities, including:

- Agents: Basics and Control Architectures
- Multi-agent Systems: Basics and Interaction
- Mobile Agents
- Interface Agents
- Coordinated Problem Solving
- Automated Decision Making
- Automated Negotiation
- Multi-agent Learning
- Logical Foundations of Agent-based Computing
- Social Simulation and Artificial Societies
- Organisation Design
- Programming and Languages for Agent Systems
- (Industrial) Standards for Agent-based Systems
- Industrial and Commercial Applications
- Computational Market Systems

Support will be available from AgentLink to enable postgraduate students from AgentLink member nodes to attend the summer school, and a special registration fee will enable other AgentLink members to participate at a reduced cost. Other participants will be welcome at the standard registration fee.

The summer school will include an "Agent Recruitment Fair", where students will have the opportunity to meet with employers active in the area of agent systems. The recruitment fair will also give employers active in agent and other advanced IT/communications industries to meet the best and brightest of Europe's graduate students.

Full details on the summer school, including the student bursary scheme and recruitment fair, will be made available soon - watch the WWW site for details!

AgentLink Support for Workshops and Conferences

At the management committee meeting of 25 September 1998, it was agreed to offer travel support for two workshops:

- Central and Eastern Europe Workshop on Multi-Agent Systems (CEEMAS'99), 30th May-3rd June 1999. St Petersburg, Russia.
- Multi-Agent Systems - Third UK Special Interest Group Workshop (formerly 'FoMAS'), Manchester, U.K. 14th-15th December, 1998

For information on how to apply for support for workshops and conferences, please contact Yves Demazeau (Yves.Demazeau@imag.fr).

AgentLink Support for Travel to MAAMAW-99

As part of its research promotion activities, AgentLink has agreed to provide some financial support for AgentLink members with full accepted papers at the MAAMAW-99 workshop. This support aims to cover the cost of registration, travel, and subsistence for the workshop. Note that support will be limited to one person per accepted paper, and will *only* be available to *AgentLink members* with *full* accepted papers (i.e., *not* posters). Further details are available on request from coordinator@agentlink.org.

Details about MAAMAW-99 are available from Magnus Boman (mab@dsv.su.se).

Second SIG Meetings

The second meetings of AgentLink's SIGs will be held in London, UK, on 21 and 22 April 1999, and will be co-located with the PAAM-99 conference. A large number of free registrations for SIG participants have been negotiated with the PAAM organisers. Contact SIG coordinators for details on these SIG meetings.

AgentLink Continuation Funding

The fifth framework looms ever closer, and AgentLink must apply for continuation funding pretty much as soon as the commission allows us to. We are interested to hear your suggestions for strategic directions that AgentLink should go in: What should we do that we are not currently doing? What are the important topics for agent R&D that we are currently ignoring? Let us know. You can email the network coordinator directly at M.J.Wooldridge@elec.qmw.ac.uk.

What is a "Special Interest Group?"

Mike Wooldridge
AgentLink, Queen Mary and Westfield College
M.J.Wooldridge@qmw.ac.uk

AgentLink has established a number of "Special Interest Groups", or SIGs for short. SIGs have essentially two purposes. The first of these is to pump-prime collaborative research and development activities in specific, strategically important sub-areas of agent-based computing. The idea is that by providing resources to bring together researchers and developers who share an interest in a particular aspect of agent systems, we will create a critical mass of activity, from which new collaborative projects and other activities will spring. For example, a reasonable expectation might be that a SIG would go on from AgentLink to apply for working group status under the European Fifth Framework program. SIGs will typically be focussed on application areas of interest (e.g., cooperative information systems, electronic commerce) or else on research areas of importance for agent systems (e.g., software engineering and methodologies for agent systems). SIGs are resourced to the extent of funding about 10 people to attend two to three meetings per year. One of the most important aspects of SIGs is their openness, in the sense that are far as possible, anyone is welcome to attend. All SIG meeting announcements will be made public, and as far as is reasonably possible, everyone will be welcome to attend. All SIGs have their own WWW page, (accessible via the AgentLink WWW site), and some have open mailing lists for members to keep in touch.

The second purpose of SIGs is to ensure that AgentLink is reactive to the demands of the community the network serves. Thus, rather than having strategic directions imposed from a small number of nodes, AgentLink aims to take its direction, as much as is reasonable possible, from member nodes themselves. In order to facilitate this process, we aim to get feedback from grassroots members via SIGs. Each SIG has representation on the AgentLink management committee, and SIGs will be encouraged to give input to the network through this and other mediums.

Note that it is not the intent of SIGs to create "projects within projects" or to directly fund research and development. Rather, the aim is that SIGs should act as facilitators activities.

SIG activities will all be reported in AgentLink News, and reports will in addition be made available through the AgentLink WWW site. But the best way to find out about SIGs is to participate in them! Remember, some (limited) funding is available for this, so why not take advantage of it and take part?

Agent Mediated Electronic Commerce

Carles Sierra
Artificial Intelligence Research Institute, CSIC, Bellaterra, Catalonia, Spain
sierra@iiia.csic.es

Electronic commerce is one of the application areas where there is a clear growing interest by the research community on agents and multi-agent systems. This interest is clearly explained by the intuitive metaphor that a market represents in terms of a society of interacting agents. In a market, agents are buyers, sellers, brokers and possibly market staff, that engage in dialogues and interact in specific ways. These electronic counterparts of real trading agents (human beings) may free them from tracking the market, they can help in satisfying the rules of the market and book keeping the commitments associated to the interaction within the market.

Agents may also perform different useful tasks in more open environments, like finding products satisfying a set of users' requirements (product brokering), or negotiating particular conditions for a commercial transaction on the users' behalf.

The design of such agents is not an easy task. In the case of electronic markets, the agents' 'world' is constrained, or structured, by a set of strict rules of behaviour. But even in that case, and for sure in the more open settings, there are many factors that are unknown and hence force researchers and developers to find appropriate agent architectures to model these factors. What mental concepts are needed to build agents for electronic commerce was a common concern.

The First Meeting

The number of research lines of the participants was very wide, being the most massive one that of negotiation - 6 groups showed interest on this topic. The need for specification environments was a worry and an interest of several groups. Also, the modelling of basic concepts for electronic commerce, such as trust, norms, authentification and preference modelling was pursued by different groups. Generic application domains like auctions was of interest to some of the participants. Research interests were

split into 'agent architectures' and 'multi-agent system architectures'. Areas like evolutionary computing, *fair* protocol design, coalition formation or security were used in the research on multi-agent systems and areas like preference modelling or BDI on the research on agents architectures. There were many positions on what type of agent architectures were suitable for electronic commerce. In the next point I summarise the most prominent ones.

Agent architectures The different groups in the meeting pointed out several components in an agent architecture that seemed necessary to model agents participating in electronic commerce:

- *Profile matching* The modelling of user preferences is essential if an agent is to be instructed to perform an automated mediation task. The agent will be faced to different choices and it must be able to establish preferences among them. Different techniques to model profiles, like for instance rough sets, were presented.
- *Deliberative* Different groups are following a deliberative approach to agent design, like modelling using BDI, goals, plans, persuasion. Different approaches to agent architectures were presented, for instance using case-based reasoning or fuzzy rules.
- *Models of personality* The use of psychological personalities can give good means to model strategies for agents in order to behave in a complex market setting. Emotional computing can be effective in modelling the relation between humans and their assistant agents.
- *Mobile agents* The need of giving mobility to agents was acknowledged. The research efforts concentrate on how to guarantee termination, security or exactly-once protocols. To protect agents against malicious hosts agents should contain time-limit validity, and electronic money with expiration date, a part from the need for third parties to give security to transactions.

Application domains The attendants of this first meeting showed a variety of interests in what concerns application domains. Market places, Electricity management and retailing were the most common. Other applications were on banking, telecommunications, finance, marketing and shopping assistants. Most applications were at a very initial stage of development. In some cases, researchers just showed their intention to start working in one of the previously listed application domains.

Products There were few products presented in the discussion. This shows the early stage of transference from theoretical ideas into realisations that Europe is facing. The presence of many industrials in the meeting, and their presentations, gives a hint, however, that several products are going to appear soon. There were two products presented:

- *Mole 3.0* A mobile agent system written in Java. Its functionalities have been tailored to support electronic commerce. The platform offers reliable execution of agents against failures of the underlying network system. Guarantees exactly-once execution and gives different security mechanisms to avoid malicious attacks to the mobile agents. Contact: Markus Schwehm (Markus.Schwehm@informatik.uni-stuttgart.de).
- *FM 96.5* A platform for the creation of auction houses. Current protocol is Dutch. Tools for monitoring are provided. The creation of agents is supported by program patterns in CommonLisp, C and Java. Free distribution (http://www.iiia.csic.es/Projects/fishmarket).

The company Living Systems presented their product at the SIG on Intelligent Information Agents and Cambridge consultants expressed their intention to extend their current products on electronic commerce (EasyChoice and NewsVenturer) with agents' technology.

Analysis of the First Meeting

The meeting showed a good balance between Academy and Industry (7 industrial group presentations and 8 research group presentations). There was a high level of coincidence in the needs and the challenges for the future. The number of products presented was very short. This is something that will probably change in the near future as long as there is already a dozen projects funded by the Commission involving the development of such products, and the development plans that several industries presented at the meeting. The fifth framework program will probably influence in a quicker product development.

Most of the 26 attendants presented interdisciplinary work, and pointed out lacks in their research that would profit from more interdisciplinarity, economics and social sciences should be the target communities for this. The two panels, on theory and practice, showed some sintony between the participants.

The use of agents in electronic commerce has been justified by several authors by the fact that they can free traders from trading details, can explore more possibilities and are more efficient that human traders. In order to reach this functionality, the panel on theoretical challenges focussed on a series of issues that the community has to face.

It was a quite common thought that economic models should be useful in order to build strategies for agents. However, most of the theoretical models coming from economics are not valid for multiple encounters, which is a more than probable scenario for agent-mediated electronic commerce. Inspiration should come from other fields, like social sciences. Also, a more clear connection between logic and game theory, and models for electronic commerce are missing. For instance, in the connection between the modelling of obligations, which has been undertaken in logic, and electronic commerce, where obligations are of vital importance -as for modelling contracts. The importance of producing standards and specification environments as an outcome of the research was stressed during the discussion. FIPA was considered, given its more that 50 members from over the world, a good framework for the management of standards.

The panel on practical challenges brought into discussion some of the key issues for the practical development of agent-mediated electronic commerce. Given the open environment that electronic markets will represent, agents necessarily will have to be adaptive. Also, mobility is, in the long term, an essential issue. The modelling of trust, the creation of virtual enterprises and the enormous variety of legal issues that the community has to face were also discussed by the members of the panel.

Future Work

As an outcome of the first meeting of the SIG on Agent Mediated Electronic

Commerce a series of actions were agreed by the participants:

1) The co-ordinator will do actions in order to involve in the SIG activities those groups belonging to consortia already funded by the commission. Probably the short time in preparing and distributing the information about the kick-off meeting made it difficult to have a more massive presence of such groups.

2) The web pages are going to be kept alive, and pointers to the slides of the presentations are going to be provided to the co-ordinator by each of the speakers, to increase the mutual knowledge of the participants and their research interests.

3) The green paper, whose elaboration is one of the objectives of the SIG, will start by the generation of a questionnaire to be distributed between the members of the SIG and other researchers and developers involved in the area.

4) Finally, it is planned to organise the next meeting around working groups. The four working groups that appear to be of most interest are: i) Negotiation, ii) Market and protocol specification, iii) Preference modelling, and iv) Security. Next meeting will take place in London during PAAM, 21-22 April. There will be common activities with other SIGs that will be announced shortly.

Intelligent Information Agents (I2A)

Innes Ferguson, co-coordinator
Active On-line Systems Ltd., London, UK
innes@active-online.net
(Matthias Klusch, coordinator
klusch@cs.cmu.edu)

The research and application area of Intelligent Information Agents (I2A) is of rapidly increasing importance. Information agents are computational software systems that have access to multiple, heterogeneous, and geographically distributed information sources. A common task of such agents is to perform, on behalf of their users or other agents, active searches for relevant information over both local and remote domains. These complex searches will require that agents have the ability to retrieve, filter, analyse, manipulate, and integrate the information discovered.

Information agents have to cope with the increasing complexity of modern information environments, ranging from relatively simple in-house information systems, through large-scale multi-database systems, to the much-vaunted cyberspace over the Internet. To cope with such information environments agents will have to deal with uncertain, vague or even incomplete information. Indeed, the effective handling of uncertainty is critical in designing, understanding, and evaluating computational systems tasked with making intelligent decisions.

As described in its original statement of Purpose & Goals, the main aim of this SIG is to provide a European interdisciplinary forum for industrial and academic parties involved in the research and development of intelligent information agents – and in particular, intelligent information agents in open, distributed information environments.

The SIG is interested in understanding the extent to which techniques and methods from areas such as Artificial Intelligence, Database Systems and Information Retrieval can be applied to the problem of information discovery by single or groups of information agents in the Internet or World Wide Web (WWW). Methods which have already born fruit and so are of particular interest here include: means for promoting interoperability among heterogeneous systems and sources, techniques from machine learning and evolutionary computing for adaptive filtering and user profiling, as well as a variety of symbolic and numerical approaches for reasoning about uncertainty.

A number of long-term goals have been established for the SIG. These include:

- Understanding the benefits, constraints and limitations of applying agent technology to database, multi-database and other information systems and environments.
- Investigating the impact of information agents on the everyday life of users in cyberspace. This could range from assessing the usage of today's relatively simple ShopBots or SearchBots, to investigating the impact of more sophisticated rational, utility-based, or adaptive agents in the real-world electronic marketplace of the Internet.
- Investigating the various issues involved in modelling the interactions between humans and agents, as well as between agents and agents. Techniques of particular interest are those for providing cost- and time-effective navigating and searching over large, unstructured multimedia information spaces.
- Identifying methodologies for the specification, analysis, design and implementation of different types of information agents (and systems of information agents).
- Establishing benchmarks for measuring the performance and efficiency of information agents (and systems of information agents) in dynamic, open information system environments;
- The development and prototypical implementation of novel types of information agents for the Internet or WWW.

The I2A SIG is intended to promote synergetic efforts on applying software agent technology to databases and information systems in open environ-

ments such as in the Internet and large corporate intranets. It is expected that the SIG will help to solidify existing and establish new cooperative ventures among I2A research groups; in addition, it should help focus scattered efforts in I2A and related fields.

The First Meeting

The purpose of the SIG kick-off meeting, then, was to provide an initial gathering in which I2A researchers and practitioners could exchange information, share research findings, and discuss potential collaborations and partnerships in intelligent information agents. Researchers and developers from a number of areas of activity were targeted, including Artificial Intelligence, Database and Information Systems, Information Retrieval and Information Sciences, Distributed Computing, Computer Support Cooperative Work, Formal Methods, and Human-Computer and Human-Agent Interaction.

The meeting was divided into two half-day sessions. The morning session started off with some general remarks from the chair and SIG co-coordinator Innes Ferguson, followed by a panel session on communication, co-operation and knowledge management in open environments. The panel session consisted of eight brief presentations, followed by a fairly lengthy and open discussion on issues arising from the scheduled talks. Stefano Antoniazzi (Italtel, Italy) addressed the impact of modern-day interactive multimedia appliances (such as set-top boxes) on user interface design and described work on a Java-based Web browser for home appliances. Chihab Hanachi (CERISS University of Toulouse I, France) described his group's work on active databases and the application of petri-nets to cooperative objects. Misbah Deen and Chris Johnson (University of Keele, DAKE Centre, UK) described their group's work on cooperating knowledge-based systems and their application to manufacturing, diagnostics and fault repair. Aldo Dragoni (University of Ancona, Italy) presented his group's work on evaluating the reliability of information sources and the modelling of various characteristics of these sources such as reliability, truthfulness, competence, efficiency, and relevancy. Pascal van Eck (Vrije University of Amsterdam, The Netherlands), standing in for his colleague Catholijn Jonker, described their work on information brokering agents and the role played by metadata and ontologies in helping to mediate between users and information providers. Klaus Fischer (DFKI, Germany) gave an account of the application-oriented MAS research taking place in his lab, which has already generated several applications in such areas as loading dock automation, transportation scheduling, traffic telematics, and virtual enterprises. Anna Ciampolini (University of Bologna, Italy) discussed her work on designing agent architectures and the integration of abductive logic for operating in open, multi-agent worlds. To end the presentations section of the panel, Mihail Matskin (Norwegian University of Science and Technology, Norway) described his group's work on agent architectures and their application to supporting cooperative work and electronic commerce.

The morning session was completed with demonstrations of two information brokering systems: OntoBroker by Stefan Decker from the University of Karlsruhe, and a multi-agent system for information brokering by Pascal van Eck from the Vrije University of Amsterdam.

The afternoon block started off with an invited talk titled "Economics, Markets and Rational Information Agents" by Scott Moss of the Manchester Metropolitan University. In addition to providing a theory describing the conditions under which one would want to use information brokers (rather than obtaining the information oneself), Scott addressed the question of why economics is increasingly being regarded as important to I2A, and compared the relative virtues of social versus economic modelling.

The invited talk was followed by a panel/discussion period on adaptive information agents and virtual information spaces. Wolfgang Pohl (GMD FIT-HCI Research Dept., Germany) described his group's work on information brokering, user modelling, and user-tailored information environments. Vadim Ermoyalev (Zaporozhye State University, Ukraine) presented work on a CORBA-like architecture that addresses the semantic gap between users and systems designers. Daniel Ballin (University of Salford, UK) gave an overview of his group's work on the development of avatars and synthetic information agents and their application in intelligent virtual environments. Paolo Petta (Vienna University of Technology, Austria) gave an account of his work on providing intelligent support for exploration in open-world information domains. And David Merceron (EURIWARE, France) described his group's work on applying machine learning and interface devices such as avatars to facilitate the interaction between human and software agents.

Following the panel session, two system demonstrations were given: one on electronic auctions by Kurt Kammerer from Living Systems AG and another on collaborative information gathering by Keiichi Nakata from the FIT-CSCW group at GMD. The day was completed with a group-wide roadmap/planning session.

The first I2A SIG meeting was widely regarded as a success by its 27 participants. Although an incredibly large number of topics and areas of study were represented - to be expected, perhaps, given the highly multi-disciplinary nature of the SIG - there was also much overlap in both purpose and interests. The top five among these essentially common interests included:

- Information (modelling such dimensions as its relevancy, quality, or currency; and performing actions over it such as search, discovery, and personalization);
- Agent architectures;
- Ontologies and metadata extraction and representation;
- Brokering, mediation, and interoperation (including issues pertaining to human-agent, agent-agent, and agent-legacy interfaces); and
- Databases (active, cooperative, federated).

Future Work

Most, if not all, participants expressed an interest in forming collaborations and to co-operate in one form or another. Moreover, concrete statements of interest were given to jointly propose specific projects for future EU funding (e.g. the upcoming Fifth Framework). A number of specific action items were also raised, several of which have already been accomplished or are actively being worked on. Among those already accomplished are the creation of a majordomo mailing list (infoagents@gmd.de), which interested parties are invited to submit contributions to; and the deployment of a BSCW shared workspace for the exchange and sharing of research publica-

tions, software, and SIG documents. (Contact Keiichi.Nakata@gmd.de or klusch@cs.cmu.edu if you're interested in using this facility.) Ongoing action items include promoting the SIG to related communities - in particular, HCI, CSCW, Distributed Computing, and Database Technology - and actively "recruiting" more industry-based practitioners, to balance the predominantly academic make-up of the present-day SIG.

The next scheduled I2A SIG meeting will be at the PAAM-99 conference in London, UK (April 21-22, 1999), which will be held in conjunction with AgentLink's other SIG meetings. For up-to-date information about the I2A SIG and its upcoming activities, and to obtain more detailed research reports from the participants cited above, check out the SIG's homepage at:

http://www.informatik.tu-chemnitz.de/~klusch/i2a-SIG.html.

Methodologies and Software Engineering for Agent Systems

Jan Treur
Department of Artificial Intelligence, Vrije Universiteit Amsterdam
treur@cs.vu.nl

It is crucial that the basic principles and lessons of software and knowledge engineering are applied to the development and deployment of multi-agent systems. At present, the majority of existing agent applications are developed in an ad hoc fashion - following little or no rigorous design methodology and with limited specification of the requirements or design of the agents or of a multi-agent system as a whole. To develop methods with which both the requirements on such systems and the systems themselves can be modelled and specified at a conceptually acceptable level of detail, characteristics of real-world multi-agent applications need to be identified, in relation to specific domains. Such specifications describe the semantics of systems without concern for implementation details, providing a basis for verification, validation and testing of the functionality of the systems in the light of the specified requirements. The purpose of this SIG is to focus on these issues. As such, the SIG will bring together a strong sub-community of AgentLink in an area of particular European strength generally.

The main purpose of this SIG is to provide a European, multi-disciplinary forum for all researchers and practitioners who are interested in research and development of methodologies and software engineering for agent systems. This concerns, in particular, people addressing the following topics in their research or practice:

- requirements engineering for agent systems
- analysis and design techniques for agent systems
- formal techniques for specification, design, and verification of agent systems
- specific ontologies for agent requirements and agent models
- reuse of agents and agent components
- libraries of generic models of specific types of agents and agent components
- validation and testing techniques
- tools to support the agent and multi-agent system development process

This SIG aims at exploiting synergy from the interaction between the software and knowledge engineering communities and the agent community, and has a strong emphasis on practical use in industry.

The first meeting

The aim of this kick-off meeting was to establish a detailed agenda for the SIG, mapping out the key issues that need to be addressed in order to make agent-based software development a reality, and the key interests and problems of participants. To participate, attendants sent a position statement, stating the key issues to be addressed in order to realise the vision of agent-based software engineering, and summarising own work addressing these issues.

The meeting was structured in sessions with presenters around five perspectives: industrial, component-based, interactional/organisational, programming languages & foundations, and knowledge engineering perspective. To obtain a form of unification of the outcome of the meeting, in each of the sessions the following points of attention were addressed:

1. *Design description.* Methods and techniques to obtain a *design description* of an agent system; i.e., conceptual modelling/specification languages used to describe a system design.
2. *Properties and analysis.* Methods and techniques to specify and establish (required) *properties* of an agent system and relations between these properties; e.g., as needed in requirements engineering and formal analysis or verification.
3. *Implementation*. Methods and techniques for *implementation* in the context of prototyping or of achieving an operational system.
4. *Reusability*. Methods and techniques to specify and maintain reusable models and reusable software for multi-agent systems, agents and agent components.
5. *Tool support*. For example, requirements engineering tools, design tools, prototyping tools and implementation environments, verification tools.
6. *Other*.

For each of these points, identification of existing potential: what was already achieved , and issues to be addressed: what has still to be done.

Design description In recent years various conceptual and logical modelling languages for agent systems in informal, semi-formal or formal graphical, textual, or logical form have been developed. Examples are:

- AALAADIN (University of Montpellier, interaction/organisational perspective)
- ACL, agent communication language (FIPA)
- Concurrent METATEM (Manchester Metropolitan University, temporal logic perspective)
- DESIRE (Free University Amsterdam, component-based/knowledge engineering perspective)
- MAGMA (University of Grenoble, interaction/organisational perspective)
- MAS-CommonKADS (University of Valladolid, knowledge engineering perspective)
- KSM (University of Madrid, knowledge engineering perspective)
- 3APL (Utrecht University, programming semantics/dynamic logic perspective)

A number of aspects have still to be addressed, and also a need for some form of unification of the various approaches is felt:

- UAML: a Unified Agent Modelling Language
- Substantiated extension of OO to agent-oriented SE; extending/modifying UML
- Documentation/design rationale
- Reference architectures for classes of agent system applications
- Extending logical approaches beyond simple examples, to more complex models
- Logical foundation for languages addressing more complex applications
- Extending existing languages with, e.g., ACL and dynamic agent creation
- Real-time aspects
- Languages for modelling mobility, with concepts to express, e.g., places/ locations, migrations of agents

Properties and analysis A number of approaches to specify and relate properties of agent systems in informal, semi-formal or formal form have been developed. For example:

- ALBERT II, a temporal logic specification language to formalise requirements for real-time agent systems (University of Namur)
- Compositional verification of temporal properties by mathematical proof (Free University Amsterdam)
- Automated verification of temporal logic specifications for simple models by logical proof (Manchester Metropolitan University)
- KARO, a multi-modal framework for formalisation of agent attitudes (Utrecht University)

A number of issues still have to be addressed:

- Requirements engineering methods for agent systems
- Ontologies and patterns for requirements
- Making formalisation more computational; achieving groundedness of formal approaches: most formalisations have no systematic and well-defined relation to descriptions of actual agent systems
- Standard relation between formalised properties and design description
- Emergence as dynamic compositionality
- Formal verification methods for more complex models
- Compositional verification based on logical proof
- Systematic testing methods
- Extension of single-agent attitude frameworks to multi-agent systems
- Proof methods for combinations of temporal logic and logics representing mental state (e.g. belief, knowledge)
- substantiation of properties such as mobility, security, co-ordination, heterogeneouty, and interoperability
- Translations of easy-to-use, e.g. graphical, formalisms into formalisms such as temporal logic

Implementation A large number of agent implementation environments have been developed. For example:

- Many variants of agent software; e.g., in Java
- Environments for automated executable prototype generation from a formal conceptual design description (execution environments included in the software environments of ConcurentMETATEM and DESIRE; interpreters of 3APL)
- Concurrent logic programming based toolkit: APRIL (Imperial College)

Needed is:

- A well-defined link between conceptual modelling languages and different existing implementation environments; agent oriented programming languages (graphical, textual,...) for implementing agent oriented design specifications
- Higher-level implementation languages providing agent concepts (e.g. goals)
- Compilers for interpreted conceptual or logical languages (such as 3APL)
- Implementation of ACL
- Interface software to existing robots

Reusability The following is already achieved:

- Notion of generic model (developed within the knowledge engineering community)
- A number of generic agent models
- Various generic models of tasks (developed within the knowledge engineering community)
- Generic requirements patterns (e.g., within ALBERT II)
- Reusable agents and agent components: models and software
- Library of reusable agent and agent component software (e.g., JIAC, University of Potsdam)
- Reusable verification proofs and generic proof patterns
- Reusable organisation structures (e.g., AALAADIN)
- A philosophy of constructing generic agents

Needed:

- Library of reusable agents and agent components: models and software
- Library of reusable agent capabilities
- Library of reusable organisation structures
- Library of reusable verification proof patterns
- Compositional semantics of combined approaches
- Development of compositional proof methods to support modularity
- Library of reusable knowledge about successful applications of certain agent technologies.

Tool support Achieved:

- Graphical conceptual design tools for some approaches
- Tools for automated generation of executable prototypes from formal conceptual design specifications
- Execution environments
- Verification tools for temporal logic specifications of simple models

Needed:

- Software environment to support requirements engineering for agent systems
- Software environment for verification of more complex models
- Compilers for agent oriented programming languages
- Agent development environments, incorporating full range of activities from requirements to verification and implementation.
- A debugger, in particular for distributed mobile agent systems

Other Achieved:

- Standardisation efforts (e.g., MAF/ MASIF, FIPA, and UAA)

Needed:

- Support for the development process (e.g., life cycle)
- Pragmatics of the development process: guidelines
- Methods tuned to specific application areas (manufacturing, information management, electronic commerce)
- Evolution of agent systems
- Explicitation of the difference with other Software Engineering
- When should Agent Technology be used for a specific application; explicitation of the benefits of certain agent technologies for certain application areas
- A benchmark for representative complex applications

Future work

For the short term, medium term, and long term future a number of agenda items were identified.

Short term (< 2000)

- Explication of why existing approaches such as UML are not sufficient, and of how development of agent systems is different
- What are crucial issues for development of a UAML
- What can agent technologies contribute today to today's mainstream software engineering approaches and languages such as UML, CORBA, and Java

Medium term (< 2002)

- Filling gaps in the technology
- Spectrum of methods tuned to different application domains
- Pilot studies and assessments

Long term (= 2002)

- Developing a UAML
- Grounded foundational approaches: well-defined relations between formalisations and descriptions of actual agent systems
- Practical implementations / realised agent application systems
- Influence on standardisation efforts and platform developers

For up-to-date and more detailed information see:

http://www.cs.vu.nl/~treur/SIG.meth0.html

ACTS CLIMATE Industrial Agent Workshop

"Experiences in the Usage of Agent Platforms and Design of Agent-based Applications - Is Agent Technology ready to use in the business?" will be held in Barcelona Monday April 26, 1999 before the the 5th ACTS IS&N 99 conference (April 27 - 29, 1999).

The main target of the agent workshop is to present to an industrial audience the first real results achieved after 12 month of hard work with agent technology within the ACTS CLIMATE projects.

The workshop web site containing detailed registration information is:

http://www.fokus.gmd.de/cc/ima/climate/industrial-agent-workshop

For more information about CLIMATE (mission, scope and projects) see:

http://www.fokus.gmd.de/cc/ima/climate/

FIPA Update

Donald Steiner
Siemens Corporate Technology, Munich, Germany
Donald.Steiner@mchp.siemens.de

At its 11th meeting in October 1998 in Durham, North Carolina USA, the Foundation for Intelligent Physical Agents (FIPA) approved the FIPA 98 specifications including:

- Agent Management Extension
- Agent Management Support for Mobility
- Agent Security Management
- Human/Agent Interaction
- Ontology Service
- Developer's Guide

as well as updated versions of the FIPA 97 specifications:

- Agent Management
- Agent Communication Language

The above specifications can be found on the new FIPA web site:

http://www.fipa.org

Furthermore, FIPA issued its 5th Call for Proposals for agent technologies to be addressed by FIPA 99.

1999		
ASAP'99	The First International Symposium on Agent Systems and Applications, California, USA.	26-29 Sep http://www.generalmagic.com/asap
IMS'99	Intelligent Manufacturing Systems Leuven, Belgium.	22-24 Sep http://www.mech.kuleuven.ac.be/pma/project/imswg/ims99
IJCAI'99	Sixteenth International Joint Conference on Artificial Intelligence Stockholm, Sweden.	13 July-6 Aug http://www.dsv.su.se/ijcai-99/
CIA-99	Third International Workshop, Cooperative Information Agents, Stockholm, Sweden.	31 Jul-2 Aug http://www.informatik.tu-chemnitz.de~klusch/cia.html
ATAL-99	The Sixth International Workshop on Agent Theories, Architectures, and Languages, Orlando, Florida, USA.	15-17 July http://www.elec.qmw.ac.uk/dai/atal
MAAMAW'99	Ninth European Workshop on Multi-Agent Systems Valencia, Spain.	30 June-2 July http://www.dsic.upv.es/~maamaw99
ICCS'99	Seventh International Conference on Conceptual Structures, Virginia, USA.	12-15 July http://www.ee.vt.edu/~iccs99/
ACL-99	The 37th Annual Meeting of the Association for Computational Linguistics, Maryland, USA.	22-27 June
UM'99	International Conference on User Modeling Banff, Canada.	20-24 June http://www.cs.usask.ca/UM99/
CEEMAS-99	1st International Workshop of Central and Eastern Europe on Multi-agent Systems, St. Petersburg, Russia.	30 May-3 June http://space.spiiras.nw.ru/ai/english/cfp.htm
EuroGP'99	Second European Workshop on Genetic Programming, Goteborg, Sweden.	26-27 May http://www.cs.bham.ac.uk/~rmp/eebic/eurogp99/
IWAN'99	First International Workshop Active Network, Berlin, Germany.	24-26 May http://www.fokus.gmd.de/ima/iwan99
WWW8	The Eighth International World Wide Web Conference, Toronto, Canada.	11-14 May http://www8.org/
Amstelogue'99	Amsterdam Workshop on the Semantics and Pragmatics of Dialogue, Amsterdam, Netherlands.	7-9 May http://earth.let.uva.nl/~amstelog/call.html
FLAIRS-99	Special Track, Parallel and Distributed Reasoning Disney Village, Orlando, Florida, USA.	1-5 May http://agent.informatik.uni-kl.de/denzinge/flairs99-padr.html
Agents'99	Third International Conference on Autonomous Agents, Seattle, USA.	1-5 May http://www.cs.washington.edu/research/agents99/
IS&N'99	Sixth International Conference on Intelligence in Services and Networks, Barcelona, Spain.	27-29 April http://www.ac.upc.es/isn99/
Coordination'99	Third International Conference on Coordination Models and Languages, Amsterdam, The Netherlands.	26-28 April http://www.cs.unibo.it/~coord99/
PAAM'99	Practical Application of of Intelligent Agents and Multi-Agents London, UK.	19 April http://www.practical-applications.co.uk/PAAM99
TINA'99	Telecommunications Information Networking Architecture Conference, Hawaii, USA.	12-15 April http://www.tinac.com/99/Conference/CFP.html
WECWIS'99	International Workshop on Advanced Issues of E-Commerce and Web-based Information Systems. California, USA.	8-9 April http://www.ibm.com/iac/news-wecwis-content.html
IEEE Meta-Data'99	The Third IEEE Meta-Data Conference, Maryland, USA.	6-7 April http://www.llnl.gov/liv_comp/metadata/md99/md99.html
ICEIS'99	First International Conference on Enterprise Information Systems, Setúbal, Portugal.	27-30 March http://www.est.ips.pt/iceis
AAAI'99	Spring Symposium on Intelligent Agents in Cyberspace, Stanford University, California, USA.	22-24 March http://btwebsh.macarthur.uws.edu.au/san/iac/
AAAI'99	Spring Symposium: Agents with Adjustable Autonomy, Stanford University, California, USA.	22-24 March http://tommy.jsc.nasa.gov/~korten/aaai-ss99/
IAC Workshop	First IAC Workshop on Internet Based Negotiation Technologies NY, USA.	18-19 March http://www.ibm.com/iac/news-iacneg-content.html

AgentLink - the European Network of Excellence for Agent-Based Computing

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 three is Friday March 5th, 1999. 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 Electronic Engineering
Queen Mary and Westfield College
University of London, London E1 4NS
United Kingdom
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
Donald Steiner
Siemens AG
ZT IK 6
D-81730 Munich
Germany
Email: Donald.Steiner@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
Blekinge University
372 25 Ronneby
Sweden
Email: Paul.Davidsson@ide.hk-r.se