Foundations of Multi-Agent Systems: Techniques, Tools and Theory*

M. Luck[1], M. d'Inverno[2], M. Fisher[3], and FoMAS'97 Contributors†

† FoMAS'97 Contributors: C. Castelfranchi; S. Parsons, C. Sierra and N. Jennings; D.T. Ndumu, H.S. Nwana, L.C. Lee and J.C. Collis; K. Binmore and N. Vulcan; R.M. van Eijk, F.S. de Boer, W. van der Hoek and J-J.Ch. Meyer; J. Padget and R. Bradford; C. Preist and S. Pearson

1. Department of Computer Science, University of Warwick, UK
2. School of Computer Science, University of Westminster, UK
3. Department of Computing, Manchester Metropolitan University, UK

Interest in agent-oriented technology continues to grow, both at a theoretical level and a practical level, with the UK maintaining a strong representation in the area in both academic institutions and commercial organisations. In December 1997, the Second UK Workshop on Foundations of Multi-Agent Systems (FoMAS'97), held at the University of Warwick, built on the success of FoMAS'96 a year earlier (Luck, 1997; Doran et al., 1997; d'Inverno et al., 1997; Fisher et al., 1997) in seeking to provide a forum for academics and industrialists within the UK to present and discuss current directions in research and applications development. Supported by the EPSRC and HP Labs, FoMAS'97 was expanded to two days with invited presentations from Professor Ken Binmore of UCL, and Professor Cristiano Castelfranchi of the Italian National Research Council's Institute of Psychology, in addition to paper presentations and panel discussions. The aim again was to provide an opportunity for promoting and supporting activity in the research and development of multi-agent systems.

The two panels, which were concerned with Agent Systems and Applications, and Agent Rationality, comprising leading academics and industrialists, each generated stimulating and sometimes heated debates. Reports on these panel discussions follow separately. In this report, we summarise the other contributions to the workshop through paper presentations and invited talks, which cover a wide range of relevant topics. The structure of the report reflects the organisation of the workshop.

The workshop began with an invited talk from Castelfranchi, who focussed on foundations of multi-agent systems with a social and psychological perspective. His starting point is the new AI of the '90s in which an important stream is *artificial social intelligence*. The main claims of the talk can be summarised as follows. First, the real foundation of all sociality (cooperation, competition, groups, organisation, etc.) is individual social action and mind. One cannot reduce or connect action at the collective level to action at the individual level unless one passes through the social character of individual action. Second, important levels of coordination and cooperation necessarily require minds and cognitive agents (beliefs, desires, intentions, etc.). However, cognition, communication and agreement are not enough for modelling and implementing cooperation: emergent pre-cognitive structures and constraints should be formalised, and emergent forms of cooperation are also needed among planning and deliberative agents. The final claim was that we are going towards a synthetic paradigm in AI and Cognitive Science, reconciling situatedness and plans, reactivity and mental representations, cognition, emergence and self-organisation.

*This report summarises the paper presentations at the Second UK Workshop on Foundations of Multi-Agent Systems. It is based on contributions from the presenters, edited by the workshop and session chairs.

In relation to this dialectic view of the relation between micro and macro levels (Conte and Castelfranchi, 1995) and to this synthetic paradigm, the importance of AI and multi-agent systems for social simulation and social theory was illustrated, by devoting attention to the most challenging theoretical problem of the social sciences: the problem of unplanned and non-contractual social organisation and cooperation; i.e. von Hayek's problem of *spontaneous order* and Adam Smith's problem of the *invisible hand*. This is claimed to be the same as the theory of *social functions* in anthropology and sociology in which functions install and maintain themselves thanks to, and through, agents' mental representations but not as mental representations, i.e., without being known or at least intended. How is this possible in cognitive and intentional agents? Intentions seem to make social functions superfluous for explaining behaviour. We need, in fact, a strange kind of behavior in *intentional* agents, namely a behavior that is goal-oriented, finalistic, but not goal-directed, non intentional. A possible solution to this problem is to be found in a theory of *cognitive reinforcement* in which functions are just effects of the behavior of the agents, that go beyond the *intended* effects and succeed in reproducing themselves because they reinforce the beliefs and the goals of the agents that caused that behavior.

It follows that behavior is goal-directed and reason-based, i.e., is intentional action. The agent bases its goal-adoption, its preferences and decisions, and its actions on its beliefs (as a *cognitive* agent), some effect of which is unknown or unintended by the agent. Then, in a circular causality, a feedback loop from those unintended effects reinforces the beliefs or goals that generated the actions. This *reinforcement* increases the probability that in similar circumstances (activating the same beliefs and goals) the agent will produce the same behavior, *reproducing* the effects. At this point such effects are no longer accidental or unimportant, and although remaining unintended they are teleonomically produced: that behavior exists thanks to its unintended effects, it was selected by these effects, and it is functional to them. Even if these effects could be negative for the goals or the interests of (some of) the involved agents, their behavior is *goal-oriented* to these effects. There are two cognitive reinforcement principles: *belief reinforcement*, and *goal reinforcement*. Both are reinforced in two ways: with respect to their *accessibility* (activation) and *reliability* (confirmation). The reinforcement of both the belief and the goal/plan will determine a reinforcement of that behavior in a cognitive-intentional agent. In summary, the relationship between emergence and intentionality was claimed to be the most challenging issue for both cognitive agent architectures and for the theory of social agents and social behaviour.

In the first presentation in the paper session, Parsons described work carried out in collaboration with Sierra and Jennings at Queen Mary and Westfield College, London. There are many ways of designing and building agent systems. However, the most common means is probably through an agent architecture. The role of such architectures is to define a separation of concerns — they identify the main functions which ultimately give rise to the agent's behaviour and they define the interdependencies between them. This approach to system design affords all the traditional advantages of modularisation in software engineering (Sommerville, 1992) and enables complex artifacts to be designed out of simpler components. However, one problem with much of the work on agent architectures is that it is somewhat ad hoc in nature. There is often little connection between the specification of the architecture and its implementation. This work aims to rectify this rather undesirable situation. The crux of the work is to see that Giunchiglia's multi-context systems (Giunchiglia and Serafini, 1994) can be used as a means of linking architecture directly to implementation.

When used as advocated, multi-context systems can be considered as being made up of four elements. The first element is the set of contexts, each of which corresponds to some architectural unit (in the example given in the paper there are separate contexts for an agent's beliefs, desires and intentions, and a fourth context to handle communication). The second element is the set of logics employed by

the contexts; typically different contexts use different logics. The third element is the set of sets of sentences written in the relevant logics, placed in the contexts and which make up the knowledge base of the agent. The final element is the set of bridge rules which state how information may be transfered from context to context (broadly speaking they are rules of inference with antecedents and consequents in different contexts which state what may be inferred in one context from what is provable in others). The big advantage of specifying agents in this way is that because it is possible to write theorem provers for multi-context systems (indeed such systems already exist) the agent specification may be directly executed thus bridging the gap between architecture and implementation.

Next, Ndumu described work at BT Laboratories, in collaboration with Nwana, Lee and Collis, on the ZEUS toolkit which was developed to provide a rapid-engineering environment for developers of distributed collaborative agent systems. The toolkit comprises a suite of Java classes that help users to develop agent-based applications by integrating and extending some predefined classes. The design philosophy behind the toolkit was to delineate domain-level problem-solving abilities from agent-level functionality. Thus, the toolkit provides classes that implement generic agent functionality such as communication, coordination, planning, scheduling, task execution, monitoring and exception handling, and ontology management. Developers are expected to provide the code that implements the agents' domain-level problem solving abilities. The main components of the toolkit include an agent component library, a set of visualisation tools, and agent building software. The toolkit also provides utility agents such as nameservers, and visualisers.

The agent building software implements a suite of editors that enables users to interactively create agents by specifying their required attributes through visual programming. The main editors include an Ontology Editor for defining the ontology for the domain; a Task Editor for describing the planning operators and reaction scripts of agents; a Definition Editor for defining the attributes of an agent, such as the tasks it can perform and its set of initial resources; an Organisation Editor for specifying any known relationships between agents; and a Coordination Editor for selecting the required coordination strategies for agents and for defining custom negotiation protocols. Finally, there is the Code Generation Editor that inherits code from the Agent Component Library and integrates it with data from the various editors to output Java code implementing the specified agents.

The visualisation tools are facilities of the dedicated "visualiser agent" that enables users to observe, control and debug their agents' behaviour. The current set of tools includes a society tool that shows the message interchange between agents in a society; a report tool that graphs the society-wide decomposition of tasks and the execution states of the various subtasks; a micro tool for monitoring the internal states of agents; a control tool for remotely modifying the internal states of individual agents, and a statistics tool for collating individual agent and society-wide statistics. Nwana et al. (1998a; 1998b) provide more elaborate descriptions of the ZEUS Collaborative Agent Building Toolkit.

The second day began with an invited talk in which Binmore described work on game-theoretic analyses of multi-agent systems, undertaken in collaboration with Vulcan at the ESRC Centre for Economic Learning and Social Evolution at UCL. Interacting agents will try to satisfy only their users' goals and will no longer have a notion of global utility. Agents will, therefore, find themselves in situations where they have an incentive to lie, or act tough, or exploit other strategic avenues not usually associated with machines. Although the issues described above are relatively new to the AI community, they closely resemble the models studied by economists and game theorists. The sub-field of microeconomics known as game theory is widely acknowledged to provide the best available set of tools for the design of multi-agent architectures (Rosenschein and Zlotkin, 1994). However, because existing theory is sometimes misunderstood or needs further development for computer applications, researchers have been reluctant to use game theory and those that do, tend to miss out on the more recent advances, which are likely to prove most useful in such settings, listed below.

First, economic theory makes a distinction between models in which we analyse the optimal behaviour of individuals or firms given the underlying mechanism (or rules of the game), and models in which we study optimal mechanism design, given that agents behave optimally. In the current early stages of multi-agent design these two approaches are being developed simultaneously. One cannot compare two different protocols (mechanisms) without specifying the behaviour of the interacting agents. Similarly, one cannot design optimising agents without some information about the protocols governing their interaction. Second, the revelation principle of mechanism design applies also to agent design, and so the designer of a properly engineered agent can tell his client that his programming takes care of all the strategic problems involved in bargaining optimally. Third, once each agent has the necessary data from his client, there is no need, in principle, for any simulation of the bargaining process. Game theory should provide a prediction of the outcome that would follow the use of optimal strategies that can be employed immediately. The prediction that game theorists make must constitute an equilibrium of the chosen protocol, otherwise new exploitative agents will be able to invade and disrupt the system. Fourthly, it provides a classification of interacting situations based on their extensive forms. It enables us to say that two, seemingly different, situations are strategically equivalent because they can be translated into the same game form.

Economic agents have been criticised in the past as being more like computer programs than real human beings, so it is interesting to see just how useful such models could be in the design of agents and protocols. Initial experience shows that existing models still need to be considerably modified to become useful in these new settings.

In the first paper of the paper presentations, van Eijk described work in collaboration with de Boer, van der Hoek and Meyer at Utrecht University on a new multi-agent system programming language to address the concern that existing languages do not have the expressiveness to present concurrency issues in a clear, modular structure. The reason for this lack of modularity can in most cases be found in the complications caused by the interactions with the various agent features that are being covered. This claim is supported by the fact that most of the existing multi-agent languages are not yet fully underpinned by a clear semantic description.

In order to develop a multi-agent programming language that is both well-structured and well-understood from a semantical point of view, the authors advocate an incremental approach of designing such a language and start from the programming paradigms of Communicating Sequential Processes (CSP) (Hoare, 1978) and Concurrent Constraint Programming (CCP) (Saraswat and Rinard, 1990). The former view of a distributed system is as a set of synchronous value-passing processing, while the latter is as set of asynchronous processes which access a global store of information. By generalising the concept of value-passing to a general mechanism of information-passing, it is possible to obtain a programming paradigm for multi-agent systems in which the agents interact with each other by passing information along a network of communication channels. Secondly, they interact with a global source of information, which models the agents' environment; the agents observe as well as manipulate this common environment.

In this language, agents are defined by a programmed behaviour (consisting of primitive communication actions), an expertise denoting the section of the environment that can be inspected and manipulated, and an informational attitude consisting of information on their own expertise and that of others. The operational semantics of the language is defined in terms of a transition system, which constitutes an appreciated mechanism for the formal derivation of computations. However, one possible option for future research would be the transformation of this operational characterisation to a compositional description of the language, as well trying to incorporate temporal semantic descriptions necessary to describe the on-going nature of multi-agent systems.

Like CSP and CCP, pi-calculus is a well-developed and understood language for specifying and

verifying the behaviour of distributed systems. In the next talk, rather than using it as the basis for a new agent language, Padget and his co-author Bradford at the University of Bath presented a use of the formalism to define a Spanish fish market an example of an electronic market system and to investigate its suitability in defining similar systems in general. It complements earlier work on development of prototypes of the fish market (Rodriguez et al., 1997).

The players (auctioneer, buyers, buyers' manager, sellers, sellers' manager, accountant) and scenes (admission, auction and settlement) comprising the fish market are modelled in the specification. Fish loads are presented on a conveyer belt and, while the auctioneer calls out details of the lots, the decreasing sequence is displayed on a large electronic scoreboard overhead. Buyers are issued with electronic control boxes bearing a button which, when pressed, stops the descending count and identifies which box was responsible. It is this control box which inspired the introduction of the remote control process, bringing numerous benefits with it. In particular, the auctioneer now need not be accessible to (electronic) buyers and so is not subject to interference from them. More importantly, the remote control serves as a mechanism capable of preventing manipulation of the auction by premature bidding, foot-dragging and spoofing.

There were several lessons learned in building a specification of this market place. For example, it is important to start simply and refine incrementally since the pi-calculus offers no modularisation mechanisms. Further, it is suggested that descriptions are caged in an asynchronous style, that channels should be unidirectional and for one purpose, and that channel naming should be done carefully and systematically. The experience of using the pi-calculus suggests that it is probably best applied at the level of defining the interaction architecture for a given scenario. This leads to a skeletal structure into which higher functions such as negotiation and argumentation can be introduced, but which do not then have to concern themselves with the low level aspects of communication, while offering the opportunity for verification of the foundations of the system.

As well as specifying agent interactions and the protocols between them as described above it is also necessary to consider how the most effective and efficient protocol or communication strategy can be selected from a set of alternatives. The final presentation of the session, by Preist and Pearson of HP Laboratories, was concerned with the performance of different messaging protocols in multi-agent systems.

In this work *service provision* is considered in which one agent has a task which it cannot carry out itself and must be matched with another agent, the service provider, which is able to carry out this task for it, possibly receiving some payment. A simple example is considered to isolate the form of this problem. In it there are providers of a given service, each giving the same quality of service, and receiving no payment. Each provider may be unavailable; it has a probability p of being unavailable at any given time, and is unavailable for a given number of seconds on average. A group of clients make, on average, a certain number of requests per second to these providers. The clients receive contact information from a facilitator agent, acting in recommend mode.

Several protocols are examined such as Naive Broadcast, where a client receives a list of all providers from the facilitator, and broadcasts its request to all of them. Those providers currently available reply, and the client chooses one of them. In addition Naive One to One and Informed One to One are considered and it is shown that the naive one to one protocol is more efficient than the naive broadcast for high values of p, and vice versa for low values. Since the parameters may vary with time, however, it is not possible to say in advance which of the protocols will be most efficient for a given problem. Instead, the authors propose that this decision is made dynamically by the agent community. They have demonstrated mathematically and subsequently checked experimentally that if each agent chooses the protocol that will minimise the total number of messages it sends and receives, then the community as a whole will choose the protocol which is most efficient. Thus the works suggests that

protocol choices can more effectively be made dynamically in response to a changing environment rather than being built in at design time.

In addition to these presentations, several participants took the opportunity to present ongoing work in short talks, contributing to the general atmosphere. Overall, FoMAS'97 provided an occasion for constructive and engaging presentations and discussions in a well-focussed and informal forum. The continuing success of the workshop in bringing together industrialists and academics in a fruitful way, has led to what we anticipate will be a series of annual workshops to support activity in the field, and encourage closer links. The third workshop in the series will be held in Manchester in December 1998, chaired by Michael Fisher of Manchester Metropolitan University, under the new name of the UK Special Interest Group on Multi-Agent Systems, broadening its scope. Details can be found at `http://www.dcs.warwick.ac.uk/~fomas`

References

R. Conte and C. Castelfranchi, 1995. *Cognitive and Social Action*. UCL Press.

M. d'Inverno, M. Fisher, A. Lomuscio, M. Luck, M. de Rijke, M. Ryan and M. Wooldridge, 1997. "Formalisms for Multi-Agent Systems" *The Knowledge Engineering Review*, 12(3): 315–321.

J. E. Doran, S. Franklin, N. R. Jennings and T. J. Norman, 1997. "On Cooperation in Multi-Agent Systems" *The Knowledge Engineering Review*, 12(3): 309–314.

M. Fisher, J. Müller, M. Schroeder, G. Staniford and G. Wagner, 1997. "Methodological Foundations for Agent-Based Systems" *The Knowledge Engineering Review*, 12(3): 323–329.

F. Giunchiglia and L. Serafini, 1994. "Multilanguage Hierarchical Logics (or: How we can do without modal logics)". *Artificial Intelligence* 65:29–70.

C. A.R. Hoare, 1978. "Communicating sequential processes" *Communications of the ACM*, 21(8):666–677.

M. Luck, 1997. "Foundations of Multi-Agent Systems: Issues and Directions" *The Knowledge Engineering Review*, 12(3): 307–308.

H. S. Nwana, D. T. Ndumu and L. C. Lee, 1998a. "ZEUS: An advanced tool-kit for engineering distributed multi-agent systems". In *Proc. 3rd Int. Conf. Practical Appl. Intelligent Agents and Multi-Agent Technology*, 377-391.

H. S. Nwana, D. T. Ndumu, L. C. Lee and J. C. Collis, 1998b. "ZEUS: A Tool-Kit for Engineering Distributed Agent Systems". To appear in *International Journal of Applied Artificial Intelligence* 7/8.

J. A. Rodríguez, P. Noriega, C. Sierra and J. A. Padget, 1997. "FM96.5 A Java-based Electronic Auction House". In *Proceedings of the Second International Conference on The Practical Application of Intelligent Agents and Multi-Agent Technology: PAAM'97*.

J. S. Rosenschein and G. Zlotkin, 1994. *Rules of Encounter: Designing Conventions for Automated Negotiation among Computers*. MIT Press.

V. A. Saraswat and M. Rinard, 1990. "Concurrent Constraint Programming". In *Proceedings of Seventeenth ACM Symposium on Principles of Programming Languages*.

I. Sommerville, 1992. *Software Engineering*. Addison Wesley.