Formalisms for Multi-Agent Systems*

Mark d'Inverno[1], Michael Fisher[2], Alessio Lomuscio[3],
Michael Luck[4], Maarten de Rijke[4], Mark Ryan[3] and Michael Wooldridge[5]

1. School of Computer Science, University of Westminster, UK
2. Department of Computing, Manchester Metropolitan University, UK
3. School of Computer Science, University of Birmingham, UK
4. Department of Computer Science, University of Warwick, UK
5. Agent Systems Group, Zuno, London, UK

1 Introduction

As computer scientists, our goals are motivated by the desire to improve computer systems in some way: making them easier to design and implement, more robust and less prone to error, easier to use, faster, cheaper, and so on. In the field of multi-agent systems, our goal is to build systems capable of flexible autonomous decision making, with societies of such systems cooperating with one-another. There is a lot of formal theory in the area but it is often not obvious what such theories should represent and what role the theory is intended to play. Theories of agents are often abstract and obtuse and not related to concrete computational models.

For example, formalisms such as temporal logics and multi-modal logics seem some distance from agents that have actually been implemented. All too often, a new logic, notation, or formalism is presented, without any convincing attempt to explain what its purpose is. Clearly, if we claim that theories expressed in such formalisms are agent specifications, then there must surely be an obligation to establish a clear link between the formalism and real agent applications.

This panel was concerned with several fundamental issues relating to formalisms for the description and development of multi-agent systems.

2 What role do formal theories play?

> *What role do formalisms and theories play in the agent field? Are they really specifications? If so, can you go from specification to implementation, and if not, what use is a formalism as a specification?*

The most common and familiar formalisms are logic-based, and logic in AI has primarily been used for one of three things: (1) formal philosophy – using mathematical logic to axiomatise such things as common sense reasoning (e.g., reasoning about knowledge and belief, intention, desire, know-how, and so forth), (2) knowledge representation, and (3) as a basis for programming. The ground-breaking work of Hintikka (1962) is a good example of the first usage. There are many examples of logic for knowledge representation, but the basic idea is widely attributed to John McCarthy. The third idea — logic as a basis for programming — is exemplified in the *Prolog* language.

This three-way split has, to a large extent, continued in work on agents. We see the first strand in the work of, for example, Cohen and Levesque, whose hugely influential theory of intention has acted as the catalyst for much related work, and the BDI (belief-desire-intention) theories of Rao and Georgeff. Logic

*This report is the result of a panel discussion at the First UK Workshop on Foundations of Multi-Agent Systems (FoMAS'96). All members of the panel are authors, listed alphabetically.

as knowledge representation language, for direct manipulation within an agent system, is exemplified in the work of Konolige on formalisms for modelling belief, and logic as a programming language is evidenced in the work of Fisher on Concurrent METATEM. All of these strands of work can claim some measure of success. However, a common failing of formal work (both in AI and multi-agent systems) is that its role is not clear.

Formal agent theories are agent *specifications*, not only in the sense of providing descriptions and constraints on agent behaviour, but also in the sense that one understands the term 'specification' from mainstream software engineering, namely that they provide a base from which to design, implement and verify agent systems. Agents are a natural next step for software engineering; they represent a fundamentally new way of considering complex distributed systems, containing societies of cooperating autonomous components. If we aim to build such systems, then principled techniques will be required for their design and implementation.

We aim to assist the development of such systems by providing formalisms and notations that can be used to specify the desirable behaviour of agents and multi-agent systems; a requirement is that we should be able to move in a principled way from specifications of such systems to implementations.

The properties identified by using a formalism serve to measure and evaluate implementations of agent systems. Some properties currently seem to be unimplementable, because they deal with an idealised aspect of agency, such as knowledge. Still, these properties can be useful because approximating them provides a basis for comparison of implementations. The overriding concern, of course, must be that the development of formal theories should be guided by the needs of practical applications of agents.

3 What should an agent theory or formalism look like?

> *Agents are programs, but if they are a new type of program, what is it that makes agents different from other types of program? How does this difference manifest itself in the formalisms used to represent such systems? Can existing formalisms be used to reason about agents (e.g., model oriented specification, process algebraic formalisms, program logics, temporal logics, etc.), or are extra features required?*

In general, an agent formalism should

- provide a precise and unambiguous language for specifying systems' components and behaviour;
- address the needs of practical applications of agents, by being capable of expressing some or all of various aspects of agency including, but not limited to, perception, action, belief, knowledge, goals, motivation, intention, desire, emotion, etc.
- help identify properties of agent systems against which implementations can be measured and assessed;
- measure, evaluate, classify, and study implementations.

No formalisms seem to possess all these qualities, with each providing a different emphasis. For example, in reasoning about *reactive* systems that are situated in an environment and maintain an ongoing interaction with that environment, *temporal logic* has proven to be a useful formalism. Similarly, other logics have been proposed (and have been successful) for describing particular aspects of agent systems. These include logics of knowledge and belief (Halpern and Moses, 1992) for describing agents' information states and logics for goals and intentions (Cohen and Levesque, 1990; Wooldridge, 1994a) for describing agents' pro-attitudes.

Being able to reason about agents in terms of the above components implies that we can relate a theory expressed in such terms to a concrete implementation of an agent. A major shortcoming of many

theories is that the link between a theory of an agent and an implementation of an agent is informal and intuitive. For example, in Shoham's well known agent-oriented programming proposal, he introduces a logic ostensibly intended for reasoning about agents, and then a programming language for programming agents. The logic and programming language contain the same symbols, and it is clear that the logic is somehow intended as a semantics for the programming language, but no precise link is ever made between the two. We believe that one should be able to give a precise logical semantics to an agent programming framework in just the same way that one can give a precise logical semantics to, e.g., a standard programming language.

A complete and coherent theory of agent systems needs to relate all these issues in one unifying whole, but such a theory is clearly still some way off, if at all possible. The approach of taking only a subset of the aspects of agency, therefore, as in the above examples, appears to be the best way of making progress in pursuit of this ultimate goal.

4 What should a multi-agent theory or formalism look like?

> *Multi-agent systems add another dimension to agent-oriented systems. Can single-agent formalisms be extended for multi-agent systems? What are the extra features of such systems that must be addressed, and how might this be done?*

A formalism for a multi-agent system must also deal with

- the multiplicity of agents;
- group properties of agent systems, such as common knowledge and joint intention;
- interaction among agents, such as communication and cooperation.

Some single-agent formalisms are expandable in these directions to handle multiple agents. For example, multi-modal logics are a good tool for studying multiple agents which do not interact too much, and can deal with some group properties and the flow of time. Some other logics have also emerged which attempt to deal with several aspects of agency at the same time, or address several agents (e.g., (van Linder et al., 1996; Fagin et al., 1995; Kraus and Lehmann, 1988; Wooldridge, 1996)).

However, we are still some way off having a logic which addresses all these features, and there are some drawbacks, too. One key problem with such formalisms is that, if anything, they are even further from implementations than with single-agent formalisms. Also, the macro-level, emergent behaviour exhibited by a multi-agent system does not easily lend itself to formal analysis. For these reasons, it may be that the best one can do is reason about local interactions between agents.

5 What are the current approaches, and what do they offer?

> *At least 90% of the next 700 formalisms for reasoning about agents will have no impact whatsoever on the development of the field. In view of this, what features do current approaches have that make them distinct and important for agent researchers and practitioners? What* fundamental *issues drive this work?*

Two views emerge. One is that we need to construct new techniques for reasoning about, and specifying, multi-agent systems. The other holds that we should make use of existing formalisms as far as possible, and that we can do without new techniques in many cases. In any event, however, the development of formalisms for multi-agent systems should be systematic, and we should avoid undesired proliferation of logics and other formalisms.

De Rijke argues against special tools or formalisms for agent theories as opposed to, say, object-oriented specification tools. In addition to the usual requirements (flexible and expressive tools for

evaluation, classification, measurement, etc.), obvious further aspects are modularity and scalability, as well as means for specifying the interaction between various components, but these features are not unique to agent-oriented settings, he claims.

In the panel discussion, three distinct approaches were taken.

- The adoption of well-known formal specification languages, such as Z and CSP, from traditional software engineering. While these provide specifications on the strict software engineering sense, they are less expressive with regard to mental attitudes of agents and temporal aspects.
- The use of executable (temporal) logics where specifications may be directly executed. The limited expressive capabilities of such logics are balanced by the absence of any need for a refinement process that is both time-consuming and error-prone.
- The use of modal logic as an expressive means for defining the relationships between various mental attitudes (belief, knowledge, desire etc.) despite doubt over whether they can be refined to computational models, or how complex such refinement strategies might be.

Z Model Specification

Existing software engineering techniques for formal specification, provide perhaps the easiest and most accessible formalisms. Though they are limited by a lack of expressiveness with regard to, for example, mental attitudes and temporal aspects, they are being used to significant effect. The Z specification language in particular has been adopted by d'Inverno and Luck as a means of constructing a formal agent framework (Luck and d'Inverno, 1995) within which they have been able to consider several distinct aspects of agents and multi-agent systems. Z, they argue, provides the following benefits.

1. It allows us to be precise and unambiguous in defining common concepts and terms in a readable way.
2. It is sufficiently expressive to allow a consistent, unified and structured account of a computer system and its associated operations, thus enabling alternative designs of particular agent models and systems to be explicitly presented, compared and evaluated.
3. It provides a foundation for development of new and increasingly more refined concepts. Through the use of schemas and schema inclusion, Z facilitates the description of systems at different levels of abstraction, with system complexity being added at successively lower levels. Thus, agent developers should be able to choose the level of abstraction suitable for their purpose.

Using this Z-based framework as a base, d'Inverno and Luck have developed their model to provide analyses of the social relationships between agents (Luck and d'Inverno, 1996), to provide an account of autonomous interaction (d'Inverno and Luck, 1996b), and to apply the model to existing systems as a means of evaluation and comparison (e.g., (d'Inverno and Luck, 1996a)). In addition, the framework has also served as an environment for the development of an agent-based software design methodology (Luck et al., 1996).

A criticism that has been leveled at this use of Z is that it is inappropriate for modelling interactions between agents. One way of addressing this is to model interaction from the perspective of single agents and, where necessary, use alternative formalism such as CSP to define the structures of communication protocols. For example, CSP is more appropriate for proving some properties relating to the *structure* of protocols such as absence of deadlock, while in other instances, such as specifying how the state of an agent changes in response to a message in a protocol, a state-driven Z stance is beneficial. The context and motivation are key in determining which formal representation technique is most appropriate.

The ability to switch between different formal representations is important in specifying many emerging complex technologies such as agent-based systems. In general, a multi-paradigm view of

system engineering (such as proposed by Zave and Jackson (1996)) is required where different formal methods are chosen as most appropriate for each paradigm. The important issues then become how to define consistency constraints between each paradigm model in a single unifying framework. Though this is some way off, d'Inverno and Luck argue that Z provides a well-defined platform from which we can move to CSP, modal or temporal logics, when appropriate.

Temporal Logic

Fisher and Wooldridge advocate the approach of combining temporal logic (for representing the dynamic aspects of agents) with formalisms for representing the 'cognitive state' of an agent — its beliefs, desires, and so on. A particular emphasis of their work is on giving a concrete interpretation of this formalism, in terms of actual implementations of agents.

A long-term goal of the research is to develop *formal methods* for multi-agent systems (Fisher and Wooldridge, 1997a): mathematically grounded techniques for specifying, implementing, and ultimately, verifying agent-based systems (Fisher and Wooldridge, 1997b). If used properly, they believe, logic — and in particular, modal and temporal logics — can be used to form the theoretical basis for these techniques. While there is still a considerable way to go, techniques for manipulating such combinations of temporal and cognitive notions are under development. For example, proof methods for combinations of temporal logics with knowledge and belief have been produced (Fisher et al., 1996), while execution methods for such logics are under investigation.

One aspect of this is to be able to give a precise logical semantics to an agent programming framework, stated above. Concurrent METATEM, an executable temporal logic, has been used by Fisher and Wooldridge to address these issues, and is currently the subject of a significant research effort (Fisher, 1996; Barringer et al., 1996). (More details can be found in the report from the panel on Methodological Foundations.)

Combining Logics

Rather than build a logic from scratch which addresses all the aspects of multi-agent systems, de Rijke, Lomuscio and Ryan believe that it is better to study how the existing logics dealing with individual aspects may be combined. Indeed, the field of *combining logics* is emerging as an active area, promising powerful results such as the preservation of important properties of the logics being combined (Gabbay, 1996; Sernadas et al., 1996; Kracht and Wolter, 1991; Blackburn and de Rijke, forthcoming).

The problem of combining logics is this: given two logics $\mathcal{A}$ and $\mathcal{B}$, how to combine them into a single logic $\mathcal{A} \otimes \mathcal{B}$ which extends the expressive power of each one? For example, suppose $\mathcal{A}$ addresses temporal aspects of agents and $\mathcal{B}$ addresses epistemic aspects. Their combination should be able to express both temporal and epistemic properties, but also the interaction of these two aspects: evolving knowledge, and knowledge about a changing world.

Fusion. (Kracht and Wolter, 1991) consider the fusion of two mono-modal logics, being 'the least bimodal logic containing both'. They show that several properties of the component logics transfer to the combination, such as completeness, finite model property, interpolation.

Embedding. Finger and Gabbay (1992) describe a technique in which one logic is embedded inside another. Although originated specifically for temporal logics, this technique can be generalised to any two logics. This is an asymmetric combination: the "outer logic" can talk about properties of the "inner" one, but not conversely.

Full-fibring. A more symmetric combination is provided by Gabbay's 'full-fibring' technique (Gabbay, 1996; Sernadas et al., 1996). In that work, formulas of the combined logic can mix operators from the component logics arbitrarily, and the semantics is defined by going back and forth between models of the component logics.

There are other techniques in the literature.

Lomuscio and Ryan believe these techniques are worth exploring as potential tools for building logics for multi-agent systems. Multi-agent systems theory would then aim to build formal models of agency by considering basic well-understood logics and study the possible interplays of the different components when combined together. De Rijke, however, is very skeptical about the potential of current technology to produce useful positive results on combining logics when *interaction* between concepts is involved (as with knowledge and belief).

In summary, there is general agreement about the role that formalisms should play in multi-agent systems, but a very healthy diversity of approaches to fulfilling that role. As with mainstream computer science, our aim is to build effective and efficient computer systems, and theory is often justified with reference to formal methods which allow us to specify and reason about these systems. With agents and multi-agent systems we can either develop new formalisms for these purposes, or we can attempt to use and adapt existing formalisms. Though the range of activity in the area is large and varied, spanning several techniques, there appears to be a consensus of opinion that whatever approach is taken, strong efforts must be made to resist the temptation to develop sophisticated but unnecessary new formalisms. Indeed, it is crucial to ensure that research in this area is guided by its impact on the field of multi-agent systems as a whole.

References

H. Barringer and M. Fisher and D. Gabbay and R. Owens and M. Reynolds, 1996. *The Imperative Future: Principles of Executable Temporal Logics*. Research Studies Press.

P. Blackburn and M. de Rijke, forthcoming. "Why combine logics?" *Studia Logica*.

Philip R. Cohen and Hector J. Levesque, 1990. "Intention is choice with commitment" *Artificial Intelligence*, 42(2-3):213–261.

M. d'Inverno and M. Luck, 1996. "Formalising the contract net as a goal directed system" In W. Van de Velde and J. W. Perram, editors, *Agents Breaking Away — Proceedings of the Seventh European Workshop on Modelling Autonomous Agents in a Multi Agent World*, Lecture Notes in Artificial Intelligence, 1038, 72–85, Springer Verlag.

M. d'Inverno and M. Luck, 1996. "Understanding autonomous interaction" In *ECAI 96. Proceedings of the 13th European Conference on Artificial Intelligence*, 529–533, John Wiley and Sons.

R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi, 1995. *Reasoning about Knowledge*. MIT Press, Cambridge.

M. Fisher, 1996. "An introduction to executable temporal logics" *Knowledge Engineering Review*, 11(1):43–56.

M. Fisher and M. Wooldridge and C. Dixon, 1996. "A resolution-based proof method for temporal logics of knowledge and belief" In D. M. Gabbay and H. J. Ohlbach, editors, *Proceedings of the International Conference on Formal and Applied Practical Reasoning*, Lecture Notes in Computer Science, 1085, Springer-Verlag.

M. Fisher and M. Wooldridge, 1997. "Towards formal methods for agent-based systems" In D. Duke and A. Evans, editors, *Proceedings of the Northern Formal Methods Workshop*, Electronic Workshops in Computing, Springer-Verlag.

M. Fisher and M. Wooldridge, 1997. "On the formal specification and verification of multi-agent systems" *International Journal of Cooperative Information Systems*, 6(1).

M. Finger and D. M. Gabbay, 1992. "Adding a temporal dimension to a logic" *Journal of Logic, Language and Information*, 1:203–233.

D. M. Gabbay, 1996. "Fibred semantics and the weaving of logics, Part 1; Modal and intuitionistic logics" *Journal of Symbolic Logic*, 61(4):1057–1120.

J. Halpern and Y. Moses, 1992. "A guide to completeness and complexity for modal logics of knowledge and belief" *Artificial Intelligence*, 54:319–379.

J. Hintikka, 1962. *Knowledge and Belief*. Cornell University Press.

M. Kracht and F. Wolter, 1991. "Properties of independently axiomatizable bimodal logics" *The Journal of Symbolic Logic*, 56(4):1469–1485.

S. Kraus and D. J. Lehmann, 1988. "Knowledge, belief, and time" *Theoretical Computer Science*, 58:155–174.

M. Luck and M. d'Inverno, 1995. "A formal framework for agency and autonomy" In *Proceedings of the First International Conference on Multi-Agent Systems*, 254–260, AAAI Press / MIT Press.

M. Luck and M. d'Inverno, 1996. "Engagement and cooperation in motivated agent modelling" In *Distributed Artificial Intelligence — Proceedings of the First Australian DAI Workshop*, Lecture Notes in Artificial Intelligence, 1087, 70–84. Springer Verlag.

M. Luck, N. Griffiths, and M. d'Inverno, 1997. "From agent theory to agent construction: A case study" In J. Müller, M. Wooldridge, and N. Jennings, editors, *Intelligent Agents III — Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages*, Lecture Notes in Artificial Intelligence, 1193, 49–63, Springer-Verlag.

A. S. Rao and M. P. Georgeff, 1991. "Modeling rational agents within a BDI-architecture" In J. Allen, R. Fikes, and E. Sandewall, editors, *Proceedings of the 2nd International Conference on Principles of Knowledge Representation and Reasoning*, 473–484. Morgan Kaufmann.

A. Sernadas, C. Sernadas and M. Ryan, 1996. "Combining Logics: Synchronising and Fibring" Research Report, Department of Mathematics, Instituto Superior Técnico, Lisbon, Portugal.

B. van Linder, W. van der Hoek, and J.-J. Ch. Meyer, 1996. "How to motivate your agents — on formalising preferences, goals, commitments" In Pierre-Yves Schobbens, editor, *Working Notes of 2nd ModelAge Workshop: Formal Models of Agents*, Sesimbra, Portugal.

M. Wooldridge, 1994. "Coherent social action" In *Proceedings of the Eleventh European Conference on Artificial Intelligence*, 279–283.

M. Wooldridge, 1996. "A logic for BDI planning agents" In Pierre-Yves Schobbens, editor, *Working Notes of 2nd ModelAge Workshop: Formal Models of Agents*, Sesimbra, Portugal.

P. Zave and M. Jackson, 1996. "Where do operation come from? A multiparadigm specification technique" *IEEE Transactions on Software Engineering*, XXII(7):508–528.