IJCAI-91 Workshop on Objects and Artificial Intelligence

Ioannis Hatzilygeroudis

■ The Objects and Artificial Intelligence Workshop was held on 25 August 1991 in conjunction with the 1991 International Joint Conference on Artificial Intelligence. The workshop brought together researchers in AI and object-oriented programming to exchange ideas and investigate possible avenues of cooperation between AI and object-oriented programming. The workshop dealt with both the theoretical and the practical aspects of this cooperation.

Today, object-oriented programming is recognized as an important and powerful programming paradigm, especially for the development of complex systems, that has also influenced many other computer science domains. AI, however, is looking for knowledge representation and programming techniques for developing complex applications and uses constructs (for example, frames) and notions (for example, classification hierarchy) that have similarities to constructs (for example, classes) and notions (for example, class inheritance) in object-oriented programming.

The one-day workshop entitled Objects and AI, held in Sydney, Australia, on 25 August 1991 in conjunction with the 1991 International Joint Conference on Artificial Intelligence, was designed to investigate the differences and possible avenues of cooperation between AI and object-oriented programming. It brought together researchers in both fields to exchange ideas and discuss issues concerning the theoretical and practical aspects of AI and object-oriented programming to explore possible cooperation between them. The workshop was organized by Daniel Bobrow (Systems Science Laboratory, Xerox Palo Alto Research Center); Jacques Ferber (LAFORIA, France); Mamdouh Ibrahim (EDS/AI Services), chair; and Mario Tokoro (SONY Computer Science Laboratory/Keio University, Japan).

The primary topics of the workshop were (1) object-oriented architectures for AI, (2) integration of objects and AI, (3) object-oriented knowledge representation, and (4) object-oriented AI applications.

Twelve papers were accepted, and 18 contributors representing research in Australia, Austria, France, Germany, Italy, Sweden, the United Kingdom, and the United States were invited to the workshop. This article concentrates on the main issues raised and the major points made during the presentations of the eight papers in the workshop's four sessions.

The workshop started with an introduction by Ibrahim in which he briefly emphasized the challenges facing research in AI and questioned whether object-oriented programming has achieved the expected impact on this field. Ibrahim posed a number of questions related to the theme of the workshop and asked the participants to address some of these questions during their talks and discussion.

The questions were the following: Are objects suitable for some but not all AI tasks? Given the object-oriented programming lack of formalisms and semantics, can we expect object-oriented knowledge representation to achieve real advances in AI? When objects are viewed as modules of knowledge or activities, how can they contribute to AI? How can concurrent object-oriented programming benefit AI systems? What is missing from existing object-oriented programming class hierarchies that prevents them from being used in AI applications? Are frames objects, and if so, can we consider frame languages object oriented? Is the class-instance mechanism flexible enough to represent natural operations on concepts such as ad hoc specializations? What would object-oriented languages (specifically CLOS) contribute to AI that Lisp cannot provide? How can object-oriented programming support tools be of help in designing knowledge-based systems and second-generation expert systems?

During the four sessions of the workshop, a number of researchers pointed out that most object-oriented languages and systems are not powerful enough to match AI and knowledge representation requirements, although they possess constructs and notions similar to those used in AI in general and knowledge representation in particular. They mainly lack declarativeness and flexibility. However, extended object-oriented languages and systems have been developed that are adequate to handle AI applications.

For example, object-based concurrent programming, a case of object-oriented programming that has a number of similarities with distributed AI, does not satisfy distributed AI requirements because it lacks representation, communication, and organization. Therefore, Ferber, speaking on behalf of Jean-Pierre Briot (LITP, France) and Les Gasser (LAFORIA, France), proposed a number of extensions to the object-based concurrent programming paradigm to close the gap with distributed AI, such as the introduction of more powerful object representations, a social theory of interaction among agents, a more declarative and flexible description of the behavior of objects, and strong models of organization to structure coordination between agents.

Also, class-based languages, a representative of object-oriented programming that has similarities with frame-based representations, do not match knowledge representation requirements because of the inflexibility of the class-instance model and their procedural orientation. Ioannis

Hatzilygeroudis (University of Nottingham, United Kingdom) presented a knowledge representation framework integrating logic and objects, where the object-based component is an extension of the class-based model in two respects. First, the class-instance model is extended to allow for redefinitions of methods within an instance. Second, declarative facets (as in frames) are incorporated within an object description. Furthermore, logic is used to declaratively describe methods, so that message passing plays a central role in reasoning.

Additionally, notions and techniques from AI programming languages have been used to improve object-oriented programming languages, making them adequate for AI application implementation. Chi-Ping Tsang (University of Western Australia) explained how the notion of continuation, as used in the Lisp-based SCHEME language, can be combined with object-oriented programming with the use of dynamic (unnamed) objects. This combination results in great flexibility in the searching and comparison of object trees.

Deductive object-oriented databases were also discussed at the workshop. Improvements in such systems have included the introduction of knowledge representation notions into their framework. Bernhard Nebel (DFKI, Germany) described such an improved deductive object-oriented database data model extended in two ways: First, the conjunction operator was introduced, which allows one to express multiple inheritance as part of a class description. Second, a distinction between primitive and derived classes was made, corresponding to the distinction between primitive and defined concepts in terminological knowledge representation languages, which gives classification capabilities to the model.

However, knowledge representation formalisms can improve their performance and representation by incorporating procedural object-oriented notions and techniques. For example, Dickson Lukose (Deakin University, Australia) presented *actor graphs*, a combination of the conceptual graph knowledge representation formalism and actors, where a set of methods and a message handler are attached to the declarative framework of a conceptual graph. Thus, an actor graph can respond to a message based on the method that is executed.

Also, object-oriented notions and techniques have proved useful in modeling and implementing complex AI applications. Real applications require comprehensiveness and performance, qualities not offered by knowledge-based systems.

Vladimir Bacvanski (Aachen University of Technology, Germany) suggested an integration of rule-based and object-oriented programming paradigms, where object-oriented notions and techniques are used to model the architecture of an application as well as the construction of the rule-based systems involved. A rule-based system is modeled as a framework of classes describing the behavior of the system building blocks. This integration resulted in the provision of a multiparadigm architecture that diminishes the impedance mismatch between knowledge-based and non–knowledge-based systems.

Object-based concurrent programming notions and constructs have also been used to model architectures of intelligent coordination between application modules, regarded as agents. Jean-Marc Loingtier (Framentec, France) presented ALAN, an actor-based agent language to be used for the implementation of various architectures on top of existing application programs (expert systems) to implement intelligent coordination between them. ALAN instantiates Gul Agha's (University of Illinois) notion of a transaction as a concrete object—a multiset of partially ordered events, thus unifying the representation of the local and remote activities of an agent.

Finally, object-oriented programming languages and, specifically, CLOS have contributed to the implementation of complex AI applications. James Kelly (Bolt Baranek and Newman) stressed the extensive use of object-oriented programming, especially CLOS, in the construction of the automated issue identification system (AIIS), a rule-based expert system for the Internal Revenue Service. For example, the multiple-method dispatch capability of CLOS was used to provide a straightforward mechanism for implementing threshold values that can vary, depending on a taxpayer's business, income level, and other factors. However, a number of hard issues, as a result of the introduction of object-oriented notions, still had to be solved.

It was clear from the talks that the object-oriented programming paradigm has already contributed to a number of advances in AI, mainly through its software-engineering capabilities. However, it was also clear that object-oriented approaches are not appropriate for all AI tasks, especially where knowledge representation is concerned. Knowledge representation requires the ability to be declarative and flexible, which is not offered by existing object-oriented approaches. For example, the class-instance model was proven to be too inflexible to cover all types of specialization used in knowledge representation. Also, AI classification problems are difficult to implement with existing class-based languages. Thus, although there is much effort devoted to objectifying AI applications, still more research is needed to address the role that objects can play in knowledge representation as well as AI in general.

Acknowledgment

I would like to thank Mamdouh Ibrahim for his support in writing this article and Peter Patel-Schneider for his suggestions in revising the initial version.

Ioannis Hatzilygeroudis received his first degree in mechanical and electrical engineering from the National Technical University of Athens, Greece, in 1979 and his M.Sc. and Ph.D. from the University of Nottingham, United Kingdom, in 1989 and 1992, respectively. He is currently a postdoctoral researcher at the University of Patras, Department of Computer Engineering and Informatics, Greece. His main area of research is hybrid knowledge representation systems, especially those integrating logic and objects.