Preface

Today, in 2007, the field of study known as multi-agent systems has been in existence for more than 25 years. However, only during the mid-1990s did the field begin to draw widespread attention as the hype-curve approached its zenith. Now as the first decade of the 21st century begins to wane, we find the field ever more active with branches in a myriad of disciplines and aspects of multi-agent systems theory and engineering in use throughout multiple application domains and business sectors. However, one important aspect of multi-agent systems that still lacks complete and proper definition, general acceptance and practical application, is that of modeling, despite the substantial efforts of an active research community.

In short, software agents are a domain-agnostic means for building distributed applications that can be used to create artificial social systems. Their facility for autonomous action is the primary differentiating property from traditional object-oriented systems and it is this aspect that most strongly implies that standard UML is insufficient for modeling multi-agent systems.

The focus of this book is thus on an approach to resolving this insufficiency by providing a comprehensive modeling language designed as an extension to UML 2.0, focused specifically on the modeling of multi-agent systems and applications. This language is AML—the Agent Modeling Language—the design of which is informed by previous work in this area while explicitly addressing known limitations relating to managing complexity and improving coverage and comprehension.

But why modeling in the first place, and moreover why model multi-agent systems? Software modeling is now a pervasive technique used to simplify the view of a software system by offering abstracted perspectives and hiding non-essential details. In fact one of the key benefits of traditional object-oriented modeling is the availability of easy-to-use semi-formal modeling languages such as UML. As a result, almost all contemporary software development processes make use of UML modeling for visually documenting aspects of requirements capture, software analysis, design and deployment. And so given that multi-agent systems are essentially a paradigmatic extension of object orientation principles, it is clear that a properly formulated extension

to UML is feasible for software agent modeling. AML offers such an extension by addressing the particular characteristics associated with multi-agent systems including entities, behavior abstraction and decomposition, social aspects, mental aspects, communicative interactions, observations and effecting interactions, mobility, deployment, services and ontologies.

But the availability of AML alone is not in itself sufficient to assist adoption; methodological support is also a critical concern. Therefore AML is supported by a software development methodology called ADEM (Agent DEvelopment Methodology) which provides guidelines for the use of AML in general software development processes. ADEM follows the Situational Method Engineering approach to provide flexibility in defining concrete methods customized to specific conditions of particular system development projects. One other critical aspect of modeling language utility is tool support, without which widespread adoption is essentially an unobtainable goal. In recognition of this, AML is supported by implementations of its UML profiles for several well-known UML CASE tools including IBM Rational Rose, Enterprise Architect and most recently, StarUML.

As AML is still quite new, adoption is still in an early phase but it is hoped that the public availability of the specification, methodology, tools and of course this book will help encourage widespread use. Currently Whitestein Technologies is at the vanguard of adoption having recognized a strong business need for AML and thus supported its creation and evolution. Whitestein, as a leading vendor of multi-agent systems applications, now actively employs AML throughout their product lines spanning telecommunications, logistics and business process management.

As a final note, the objective of this book is to familiarize the reader with the Agent Modeling Language by explaining its foundation, design principles, specification, and usage. The book however makes no attempt to educate the reader in the techniques of software modeling and thus a reasonable understanding of object-oriented design principles and especially UML is recommended.

Acknowledgements

Very special thank goes to Dominic Greenwood, not only for the precious discussions and advice which have inspired and shaped many ideas, but also for his generous and unfailing help concerning language refinements and wording polishing. Without his encouragement this book would hardly be published.

We are indebted also to the AML technical reviewers, namely Stefan Brantschen, Monique Calisti, and Giovanni Rimassa, for their fruitful comments and suggestions which have substantially influenced the current version of AML.

We would also like to thank professor Branislav Rovan for his encouragement and valuable contribution to the form and content of the work.

Thanks to Whitestein Technologies for their support and provision of sufficient time, team of professionals, and other resources which we utilized in our work.

We are also obliged to Hilary Greenwood for her thorough proof reading of the final text resulting in substantially improved legibility and understandability.

Our thank belongs also to the members of the FIPA Modeling and Methodology technical committees and AgentLink AOSE technical forum groups for the discussions about various topics of agent-oriented modeling and agent-oriented software engineering.

Last, but not least, to our families for their love, unconditional support, and patience.