

A Tool to Facilitate Agent Deliberation

Daniel Bryant, Paul Krause, and Sotiris Moschoyiannis

Department of Computing, University of Surrey, Guildford, GU2 7XH. UK
{d.bryant, p.krause, s.moschoyiannis}@surrey.ac.uk

Abstract. In this paper we present a prototype of a tool that demonstrates how existing limitations in ensuring an agent's compliance to an argumentation-based dialogue protocol can be overcome. We also present the implementation of compliance enforcement components for a deliberation dialogue protocol, and an application that enables two human participants to engage in an efficiently moderated dialogue, where all inappropriate utterances attempted by an agent are blocked and prevented from inclusion within the dialogue.

1 Introduction

Autonomous software agents are often cited as a key enabling technology for the next generation of distributed service provision, such as large-scale electronic commerce systems [1] and Service-Oriented Computing [2]. Key characteristics of such services are agent heterogeneity, conflicting individual goals, limited trust and a high probability of non-conformance to specifications [3]. If this vision of large-scale open multi-agent systems is to be realised then the fundamental problem of interoperability (i.e. communication between agents) must be addressed. As a result, there has been much work on agent communication languages (ACLs), and an increasing amount of this work has concentrated on argumentation-based dialogue [4]. However, for an ACL to truly be an enabling technology, it must rely on a standard or protocol to ensure that different implementations preserve the ACL's meaning [5], and in order to gain acceptance, particularly for sensitive applications such as electronic commerce, it must be possible to determine whether or not any system that claims to conform to an ACL protocol actually does so [5], [6].

In this paper we present a prototype of a tool that demonstrates how existing limitations in ensuring an agent's compliance to an argumentation-based dialogue protocol can be overcome. Dialogue protocols are enforced by means of a series of distributed "Dialogue Manager" enforcement components, implemented as a lightweight Java-based agent proxy. Our ultimate goal is to implement a generic ACL enforcement tool, but in order to keep this paper focused we will concentrate on the implementation of enforcement for a deliberation dialogue protocol, as presented in [7]. The remainder of this paper is structured as follows. First, we present an overview of the deliberation dialogue and dialogue games. Next, we summarise the implementation of our tool. We conclude the paper with an overview of the planned future work.

2 Deliberation and Dialogue Games

Hitchcock *et al* [7] state that a deliberation dialogue arises with a need for action in some circumstance. In general human discourse, this need may be initially expressed in governing questions which are quite open-ended, as in *where shall we go for dinner this evening?* In [7] a formal and implementable model for deliberation dialogues between autonomous agents is presented, utilising an ACL with argumentation-based social semantics [5] within a formal dialogue game. Formal dialogue games are games in which two or more participants "move" by uttering locutions, according to certain pre-defined rules (see [7] for a more detailed presentation). For each locution type in the deliberation dialogue a series of pre- and post-conditions are specified based on external observable information such as the previous utterances of each agent and current dialogical commitments. These conditions are used to axiomatise behaviour in the sense that they specify when the utterance of each locution would be considered a legal move in the dialogue.

3 Implementing the Dialogue Manager

There are many examples of existing work for verifying agent specifications and protocol compliance (which will not be cited due to space restrictions in this paper). However, many techniques rely on access to an agent's internal state (which is considered unacceptable to many researchers), are only capable of verifying the design level of an agent, or have not been practically implemented. There has also been several recent approaches to enforcing agent interaction that seek to overcome these limitations, most notably Artikis *et al's* *Society Visualiser* [3] and Alberti *et al's* *SOC-SI* [8]. Our work differs from these approaches in two fundamental ways. Firstly, we focus exclusively on enforcing argumentation-based dialogue protocols (which naturally contain a form of social semantics [5]) and as such we provide an efficient technique for translating locution pre- and post-conditions into executable code (based on formal support provided in [9] to represent locution conditions in a common format). Secondly, our enforcement mechanism has been distributed across all the participating agents, reducing the potential performance bottleneck of a monolithic mechanism.

We have implemented our tool in the form of a lightweight Java application using Sun's distributed JavaSpaces technology to act as the communication medium. At the core of the JavaSpaces system the "Linda-like" [10] tuples-based associative black board coordination model is utilised, decoupling the communicating agents both spatially and temporally. We have created a client-side "Dialogue Manager" proxy that acts as a mediator between every agent involved in a dialogue and the communication medium (based on the Controller in the LGI model [10]). We have also implemented the rules for the deliberation dialogue protocol and the pre- and post- conditions for each locution's semantics (as specified in [7]) using a flexible framework which is cleanly separated from the Dialogue Manager (analogous to the Law in LGI). This enables different dialogue protocols to be swapped and enforced at run time, and in future versions

of the tool will allow a variety of dialogue-types to be mediated. A Dialogue Manager operates essentially as follows: It intercepts all utterances that the associated agent attempts to make and, based on its own local copy of the dialogue rules and local control state (previous utterances and dialogical commitments), determines whether the locution would be appropriate at this time, blocking any inappropriate utterances from inclusion within the dialogue.

An additional client-side GUI tool has been created (Figure 1) that utilises the Dialogue Manager component so that a dialogue between two (geographically distributed) human participants can be undertaken under the protocol, with each participant taking turns to utter a locution. If a participant attempts to make an illegal move then they are informed accordingly and given the opportunity to choose an alternative move. All previous utterances and the current commitment store are displayed in the GUI and are publicly available to all agents participating in the dialogue (Figure 1). This facilitates the expedient resolution of the dialogue by allowing participants to determine which of their commitments overlap or conflict with those of other participants, and thereby identify points of agreement or determine which commitments are susceptible to an attack.

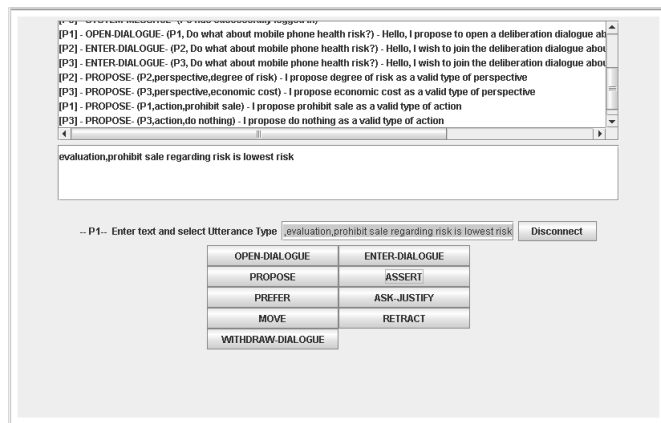


Fig. 1. Screen shot of the GUI tool allowing human participants to engage in a deliberation dialogue.

4 Conclusion and Future Work

We have presented a prototype tool that demonstrates how existing limitations in ensuring an agent's compliance to an argumentation-based deliberation dialogue protocol can be overcome. Our current application utilises a flexible protocol enforcement framework, which blocks any inappropriate or illegal utterances,

and does not require central control. We have also presented a GUI application that enables two human participants to engage in a moderated dialogue. Future work will focus on enhancing our application to support a dialogue framework in which more than one kind of dialogue can be carried out (as presented in [4]). As part of this work we are currently investigating the use of a vector language (used to model component interaction in [11]) which we believe will offer a generic representation of argumentation-based dialogues in which it is possible to capture the dependencies between moves of all the participants at each step of a dialogue.

This work was partially supported by the EU IST/STReP ASPIC project, Grant 002307, and an EPSRC PhD Studentship.

References

1. C. Guilfoyle, J. Jeffcoate, and H. Stark. *Agents on the Web: Catalyst for E-Commerce*. Ovon Ltd. London, 1997.
2. M. P. Papazoglou. Service-Oriented Computing: Concepts, characteristics and directions. In *WISE '03: Proceedings of the Fourth International Conference on Web Information Systems Engineering*, page 3, Washington, DC, USA, 2003. IEEE Computer Society.
3. A. Artikis, J. Pitt, and M. Sergot. Animated specifications of computational societies. In *AAMAS '02: Proceedings of the first international joint conference on Autonomous agents and multiagent systems*, pages 1053–1061, New York, NY, USA, 2002. ACM Press.
4. L. Amgoud, M. Caminada, P. McBurney, H. Prakken, and G. Vreeswijk. Final Review and Report on Argumentation System. Technical Report ASPIC Deliverable 2.6, 2006.
5. M. P. Singh. Agent communication languages: Rethinking the principles. *IEEE Computer*, 31(12):40–47, 1998.
6. M. Wooldridge. Verifiable Semantics for Agent Communication Languages. In Y. Demazeau, editor, *Proceedings of the Third International Conference on Multi-Agent Systems (ICMAS'98)*, pages 349–356, Paris, France, 1998. IEEE Press.
7. D. Hitchcock, P. McBurney, and S. Parsons. A Framework for Deliberation Dialogues. In *Proc. of 4th Biennial Conf. Ontario Society for the Study of Argumentation (OOSA)*, 2001.
8. M. Alberti, D. Daolio, P. Torroni, M. Gavanelli, E. Lamma, and P. Mello. Specification and verification of agent interaction protocols in a logic-based system. In *SAC '04: Proceedings of the 2004 ACM symposium on Applied computing*, pages 72–78, New York, NY, USA, 2004. ACM Press.
9. S. Wells and C. Reed. Formal dialectic specification. In *Proceedings of First International Workshop on Argumentation in Multi-Agent Systems (ArgMAS 2004)*, LNCS, pages 31–43. Springer Berlin, 2004.
10. N. H. Minsky and V. Ungureanu. Law-governed interaction: a coordination and control mechanism for heterogeneous distributed systems. *ACM Transactions on Software Engineering and Methodology*, 9(3):273–305, 2000.
11. S. K. Moschogiannis. *Specification and Analysis of Component-Based Software in a Concurrent Setting*. PhD thesis, University of Surrey, 2005.