ABSTRACT

We have been developing a framework for executable specification of norm-governed multi-agent systems. In this framework, specification is a design-time activity; moreover, there is no support for run-time modification of the specification. Due to environmental, social, or other conditions, however, it is often desirable, or even necessary, to alter the system specification during the system execution. In this chapter we extend our framework by allowing for “dynamic specifications”, that is, specifications that may be modified at run-time by the members of a system. The framework extension is motivated by Brewka’s “dynamic argument systems”—argument systems in which the rules of order may become the topic of the debate. We illustrate our framework for dynamic specifications by presenting: (i) a dynamic specification of an argumentation protocol, and (ii) an execution of this protocol in which the participating agents modify the protocol specification.
INTRODUCTION

A particular kind of Multi-Agent System (MAS) is one where the member agents are developed by different parties, and where there is no direct access to an agent’s internal state. In this kind of MAS it cannot be assumed that all agents will behave according to the system specification because the agents act on behalf of parties with competing interests, and thus they may inadvertently fail to, or even deliberately choose not to, conform to the system specification in order to achieve their individual goals. A few examples of this type of MAS are Virtual Organisations, electronic marketplaces, argumentation (dispute resolution) protocols, and negotiation protocols. MAS of this type are often classified as ‘open’.

We have been developing executable specifications of open MAS (Artikis, 2003; Artikis, Sergot & Pitt, 2003; 2007); we adopt a bird’s eye view of these systems, as opposed to an agent’s own perspective whereby it reasons about how it should act. Furthermore, we view agent systems as instances of normative systems (Jones & Sergot, 1993). A feature of this type of system is that actuality, what is the case, and ideality, what ought to be the case, do not necessarily coincide. Therefore, it is essential to specify what is permitted, prohibited, and obligatory, and perhaps other more complex normative relations that may exist between the agents. Amongst these relations, we place considerable emphasis on the representation of institutionalised power (Jones & Sergot, 1996) — a standard feature of any norm-governed system whereby designated agents, when acting in specified roles, are empowered by an institution to create specific relations or states of affairs (such as when an agent is empowered by an institution to award a contract and thereby create a bundle of normative relations between the contracting parties). We encode specifications of open MAS in executable action languages from the field of Artificial Intelligence (Giunchiglia, Lee, Lifschitz, McCain & Turner, 2004; Kowalski & Sergot, 1986).

Our executable specifications may be classified as ‘static’, in the sense that there is no support for their run-time modification. In some open MAS, however, environmental, social or other conditions may favour, or even require, specifications modifiable during the system execution. Consider, for instance, the case of a malfunction of a large number of sensors in a sensor network, or the case of manipulation of a voting procedure due to strategic voting, or when an organisation conducts its business in an inefficient manner. Therefore, we present in this chapter an infrastructure for ‘dynamic specifications’, that is, specifications that are developed at design-time but may be modified at run-time by the members of a system. The presented infrastructure is motivated by Brewka’s ‘dynamic argument systems’ (Brewka, 2001) — argument systems in which, at any point in the disputation, participants may start a meta level debate, that is, the rules of order can become the current point of discussion, with the intention of altering these rules.

Our infrastructure for dynamic specifications allows protocol participants to alter the rules of a protocol \( P \) during the protocol execution. \( P \) is considered an ‘object’ protocol; at any point in time during the execution of the object protocol the participants may start a ‘meta’ protocol in order to decide whether the object protocol rules should be modified: add a new rule-set, delete an existing one, or replace an existing rule-set with a new one. Moreover, the participants of the meta protocol may initiate a meta-meta protocol to decide whether to modify the rules of the meta protocol, or they may initiate a meta-meta-meta protocol to modify the rules of the meta-meta protocol, and so on.

We chose an argumentation protocol based on Brewka’s reconstruction of a theory of formal disputation to illustrate our infrastructure for dynamic specifications: the object and meta protocols are all argumentation protocols. In other words, at any time during a debate the agents may start a meta level argument to change the rules that govern their debate. The argumentation protocol was chosen for