Norm Compliance of Protocols in Electronic Institutions*

[Abstract]

Huib Aldewereld
Institute of Information & Computing Sciences
Utrecht University
The Netherlands
huib@cs.uu.nl

Javier Vazquez-Salceda
Institute of Information & Computing Sciences
Utrecht University
The Netherlands
javier@cs.uu.nl

Frank Dignum
John-Jules Ch. Meyer
Institute of Information & Computing Sciences
Utrecht University
The Netherlands
dignum@cs.uu.nl, jj@cs.uu.nl

ABSTRACT

There is a wide agreement on the use of norms in order to specify the expected behaviour of agents in open MAS. However, in highly regulated domains, where norms dictate what can and cannot be done, it can be hard to determine whether a desired goal can actually be achieved without violating the norms. To help the agents in this process, agents can make use of predefined (knowledge-based) protocols, which are designed to help reach a goal without violating any of the norms. But how can we guarantee that these protocols are actually norm-compliant? Can these protocols really realise results without violating the norms? In [1] we introduce a formal method, based on program verification, for checking the norm compliance of protocols.

Categories and Subject Descriptors

F.4.3 [**Mathematical Logic and Formal Languages**]: Formal Languages

General Terms

Theory, Verification

Keywords

Protocol verification, formal logic, normative systems

1. INTRODUCTION

Agents in open multiagent systems are sometimes as diverse as humans, as heterogeneous agents may behave in different ways in trying to complete their specified tasks. As some of this behaviour might not be desired, one needs mechanisms to constrain the behaviour of the agents joining the system by defining what is right and wrong. By doing so one can guarantee a safe and regulated environment for the agents to work in.

An Electronic Institution (eInstitution) is such an environment, where the expected behaviour of the agents joining the institution is described by means of an explicit specification of *norms* [2] [8]. As in human institutions, norms in eInstitutions are stated in such a form that allows them to regulate a wide range of situations over time without the need for modification. To achieve this stability, the formulation of norms abstracts from a variety of concrete aspects [4] [8]; i.e., norms are expressed in terms of concepts that are kept vague and ambiguous on purpose [5].

Because of their abstract nature, norms tend to be hard to understand and, as in real life, adhering to the norms that regulate the institution of which you are a part can be, at the least, a bit challenging. It is not unlikely that in highly regulated systems agents (and humans alike) might become overly cautious, trying not to violate any of the norms and thereby seriously reducing their efficiency and even influence the outcome and success of their goals, i.e., desired results, that *are* possible to achieve, might not be achieved anymore because the agent believes that performing the actions leading to the desired result could be violating the norms. In order to help agents act in such an environment and increase their efficiency as well as their chance of success one can specify *norm-compliant protocols* for the tasks that are to be accomplished in the institution.

A norm-compliant protocol is a guideline that makes sure that, when followed, one does not violate any of the norms, and as such it provides a quick and efficient manner to do the tasks one is assigned, since one does not need to review the norms and check norm compliance whenever one is planning to perform an action. In order to guarantee this the norm compliance of the protocol should be checked, which means that one should check that no norms are violated by the protocol during its execution in all situations. Therefore, the protocol should provide a violation-free path to achieve the agent's goals. As long as the protocol is followed *to the letter* the agent should stay out of harm's way.

In [1] we present a formal method for checking the norm

*This research was supported by the Netherlands Organisation for Scientific Research (NWO) under project number 634.000.017

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.
AAMAS'05, July 25-29, 2005, Utrecht, Netherlands.
Copyright 2005 ACM 1-59593-094-9/05/0007 ...$5.00.

compliance of protocols based on temporal logic, using an approach used in concurrent programming [6]. We have chosen this approach over traditional techniques for verifying (sequential) programs, because verification methods for concurrent programs and temporal logics allow us to check whether norms are violated in intermediate steps as well, where traditional techniques would only allow for checking the input and output of a protocol. The formalism of [6] is, however, limited to checking properties and assertions for concurrent programs, not for checking norm compliance. Therefore we enhanced the formalism with the means to express norms and violations and prove the non-violating of these norms by the protocol. For instance, to tackle the problem of linking abstract norms to the concrete steps in the protocol we had to introduce an *counts as* operator, which is inspired by [3].

The idea is that we translate the protocol into a program. The norm compliance of the protocol, being the fact that no norms are violated during the protocol run, can then be verified by proving an invariance property of the program, i.e. if we can prove that non-violation is an invariance of the program, we have shown that the protocol is norm-compliant.

The example problem that we use in [1] to demonstrate the effectiveness of this technique is a real-life protocol that describes which steps should be taken by a doctor to determine whether he can extract the organs of a donor or not (for the use of transplantation). A simplified version of this protocol is included in figure 1. We are using this real-life protocol because of the complexity of the norms applicable to the domain. We feel that if the formalism is able to express and handle such norms, it can be applied to all sorts of normative domains.

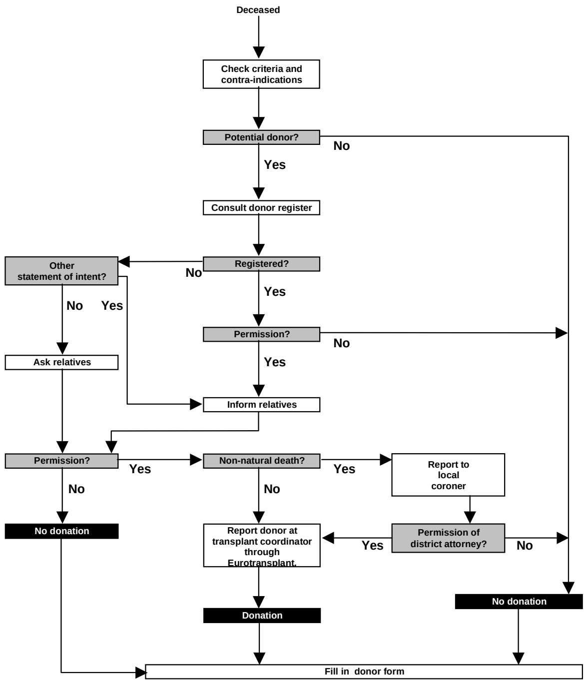

Figure 1: Protocol for organ donation.

2. CONCLUSIONS

In [1] we discuss a formal approach on norm compliance of protocols based on the verification of programs. We give a view of how these techniques can be used, after some adaptation and extention, to verify that a protocol is norm-compliant. We show, as an example, how norm compliance of a knowledge-based protocol, which is actually used in the medical domain, can be proven.

Please note that norm compliance of the protocols used by the agents is only a step towards the implementation of norms in MAS. Protocols are guidelines and agents are, therefore, not necessarily constrained to follow the protocol. A more direct enforcement is needed instead. Norms can be enforced either by the use of violation detection and sanctioning these violations [7], or by directly constraining the agents such that they can only do actions that do not violate norms.

Currently our formal method is suited for verification of single sequential protocols. We plan to extend our logical formalism to prove norm compliance of parallel protocols (such as interaction protocols). We also plan to extend the language with operators from epistemic logic in order to improve expressiveness of knowledge and beliefs of agents following a protocol. Moreover, we are very interested in seeing how this extended approach can, for instance, be used for the checking of security and authentication protocols.

The framework discussed in this paper uses a theorem proving method to verify the norm compliance of protocols. This is known to be labour-intensive. We are currently considering the use of model-checking, instead.

3. REFERENCES

[1] H. Aldewereld, J. Vázquez-Salceda, F. Dignum, and J.-J. Meyer. Proving norm compliancy of protocols in electronic institutions. Technical Report UU-CS-2005-010, Institute of Information and Computing Sciences, Utrecht University, 2005.

[2] M. Esteva. *Electronic Institutions: from specification to development.* PhD thesis, Universitat Politèchnica de Catalunya, 2003.

[3] D. Grossi, H. Aldewereld, J. Vázquez-Salceda, and F. Dignum. Ontological aspects of the implementation of norms in agent-based electronic institutions. Accepted for the 1st International Symposium on Normative Multiagent Systems (NorMAS2005), 2005.

[4] D. Grossi and F. Dignum. From abstract to concrete norms in agent institutions. In *Proceedings of FAABS III Workshop, Washington, april*, 2004.

[5] H.L.A. Hart. *The Concept of Law.* Clarendon Press, Oxford, 1961.

[6] F. Kröger. *Temporal Logic of Programs*, volume 8 of *EACTS monographs on theoretical computer science.* Springer-Verlag, 1987.

[7] J. Vázquez-Salceda, H. Aldewereld, and F. Dignum. Implementing norms in multiagent systems. In G. Lindemann, J. Denzinger, I. Timm, and R. Unland, editors, *Multiagent System Technologies*, LNAI 3187, pages 313–327. Springer-Verlag, 2004.

[8] J. Vázquez-Salceda and F. Dignum. Modelling electronic organizations. In V. Marik, J. Muller, and M. Pechoucek, editors, *Multi-Agent Systems and Applications III*, LNAI 2691, pages 584–593. Springer-Verlag, 2003.