Applying social norms to high-fidelity pedestrian and traffic simulations

Marco Robol
Department of Information Engineering and Computer Science
University of Trento
Trento, Italy
Email: marco.robol@studenti.unitn.it

Paolo Giorgini
Department of Information Engineering and Computer Science
University of Trento
Trento, Italy
Email: paolo.giorginit@unitn.it

Paolo Busetta
Delta Informatica SpA
Trento, Italy
Email: paolo.busetta@deltainformatica.eu

Abstract—**Smart cities are founded on complex interactions among architectural and urban designs, sensors, actuators and crowds of people with their devices. In this context, simulation becomes essential to study the effects of the technology and to understand how to improve its effectiveness on the social environment. The majority of the current pedestrian and traffic simulations adopt a bird-eye view and are driven by statistical models. While this is enough in many cases, e.g. to study traffic flow under common conditions assuming average cases, it is not appropriate when a higher level of fidelity is required. Simulated people need to show both a plausible behavior and mechanisms to coordinate with human participants in a natural way. Much of this coordination happens silently and is driven by social norms, that may vary according to culture and context. In this paper, we propose an approach to represent social norms in multi-agent systems that enables implicit coordination driven by observations of others' behaviors. This is applied specifically to the case of pedestrian movement. In order to allow for a more effective participation of humans in the simulation, our approach does not use central coordinators or coordination protocol, but rather each agent takes its own decision so to make more realistic interactions. A software architecture and initial experimental results are presented and discussed.**

I. Introduction

Realistically simulating the social life mechanics of people inside a city can be useful for many reasons, such as identifying and solving possible problems of the infrastructures and the services provided to citizens or visitors. In particular, pedestrian and traffic simulation is an effective way to evaluate the need for new infrastructures and spot the locations that necessitate more attentions. Additionally, high fidelity simulators may be helpful to train personnel in charge of the security or other crucial departments.

The simulation of realistically behaving people can be very complex to be implemented, particularly when basic coordination mechanisms are the result of the application of cultural- and context-sensitive social norms that may affect the behavior of each single actor.

Current solutions for pedestrian and traffic simulation focus mostly on massive simulations ruled by statistical models. Some well known products are:

- SimWalk[1] is an advanced pedestrian simulator that offers a powerful solution for analyzing and improving pedestrian flow and crowd management issues. Every single pedestrian is modeled individually with specific goals and behaviors. SimWalk allows visualization in 2D as well as 3D. It is continuously validated through validation projects, tests and comparisons with real-world data in order to deliver an accurate tool for the analysis of pedestrian flows.
- Legion[2] is another pedestrian simulator that undergoes a validation process based on a measurement program of actual pedestrians: video footage shots in different cities covering all relevant pedestrian contexts. It provides sophisticated modeling, analysis and presentation tools for a vast array of contexts and types of project.
- AnyLogic's pedestrian modeling library[3] allows to create complex analyses using the coded behavior of their pedestrian objects. These objects follow basic rules that have been determined by theoretical studies so they move at predetermined rates, they know not to occupy the same physical space as other objects, and they adjust their distance and speed based on the congestion of the crowds around them.
- SUMO (Simulation of Urban MObility)[4] traffic simulator focuses on routing algorithms and models to generate paths of pedestrians. Interactions with vehicles are dealt with a single, simplistic collision-avoidance conservative model.

It appears that current solutions for pedestrian and traffic simulation often miss a 3 dimensional graphical representation and high fidelity of simulated people behaviors adequate for an immersive experience in a serious game. Videogames and Virtual Realitys (VRs), by contrast, support very high quality 3D representations, but lack of plausibly behaving Non Player Character (NPC).

In this paper, we propose a multi-agent approach for a highly realistic simulation of people behaviors applied to controlling NPCs of a VR. Current theoretical studies show

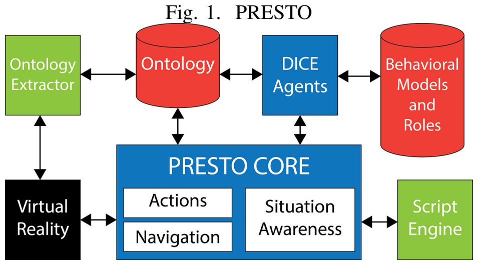
[1]http://www.simwalk.com/, SimWalk, [Accessed 20 Apr. 2016]

[2]http://www.legion.com/, Legion, [Accessed 22 Apr. 2016]

[3]http://www.anylogic.com, AnyLogic, [Accessed 22 Apr. 2016]

[4]http://sumo.dlr.de, SUMO, [Accessed 11 Jul. 2016]

Fig. 1. PRESTO



that Multi-Agent Systems (MASs) are adopted by many to study complex social systems, because the Beliefs Desires Intentions (BDI) architecture of autonomous agents is an effective way to represent rational autonomous individuals. MASs are also very useful to simulate small groups and do not necessarily focus on large crowds. In principle, MASs are more context- and culture-sensitive than specialized pedestrian simulators.

We introduce INCA, a distributed approach for the coordination of NPCs that integrate social norms into autonomous agents. We used INCA to implement implicit coordination mechanisms adopting social norms to rule the order of access to resources.

Next section presents a development suite for controlling NPCs of a VR, followed by a section on the proposed approach and another one about its implementation within the mentioned suite, which includes a scalability test. We conclude with a section about related work and final remarks.

II. Baseline - PRESTO

The work presented in this paper has been developed on top of PRESTO[5], a suite of development tools and run-time facilities for artificial intelligence in games, focusing in particular on cognitive simulation. PRESTO interfaces a number of game engines, including Unity[6], to compute the perceptions of NPCs and control their behaviour by means of autonomous agents developed with PRESTO's own DICE framework. Specifically, our work extends DICE to automatically obtain plausible behaviours when facing common situations e.g. during navigation in a 3D space.

Figure 1 shows a simplified anatomy of PRESTO. Its core provides services such as perception elaboration (Situation Awareness (SA)) and low-level control of the simulation entities. An ontology provides the conceptualization required to make behavioural models and scripts independent of the specificities of a simulation, game or rendering engine [1], [2]. A script engine controls the overall evolution of a simulation, while the DICE multi-agent framework is used for building complex NPC models as an assembly of roles, goals that can be satisfied by the latter and behavioural models implementing the tactics required to achieve goals.

[5]Plausible Representation of Emergency Scenarios for Training Operations (PRESTO), a R&D project by Delta Informatica Spa.

[6]unity3d.com, (2016). Unity Technologies [Accessed 26 Feb. 2016].

A future version of PRESTO may allow a human player to become the strategic controller of an agent. Differently from common video-games in which human players directly control gestures, postures and movements of their avatars, in PRESTO the players would give them goals. An advantage of this approach is the construction of cognitive-level GUIs (minimizing the need for HCI devices that discourage the casual player, let alone mature users that may feel uncomfortable in serious games). Internally, it allows a uniform representation and perception of humans and NPCs.

A. *Situation Awareness*

PRESTO elaborates, in real-time, the sensorial space of the NPCs to create individual streams of perceptions as ontologically classified symbols, which are processed by the controlling agents to update their internal beliefs. Perceptions represents pre-elaborated cognitive stimulus corresponding to: (i) perceived entities and their persistence in time, (ii) their features properly classified to allow symbolic reasoning.

Next versions of PRESTO will extend perceptions to include the automatic detection of situations, defined as specific patterns of locations, entities and their properties.

B. *DICE*

Delta Infrastructure for Cognition and Emotion (DICE) is a framework to implement autonomous agents capable to control NPCs of a VR in real time. It integrates the facilities of PRESTO with JACK Intelligent Agents (JACK) [3], an agent-oriented development environment built on top of, and integrated with, the Java programming language. JACK allows to develop agents based on the theoretical BDI paradigm.

DICE's internal architecture reflects the simulation cycle typical of Computer Graphic (CG) engines, which is characterized by a continuous stream of perceptions of events, to which PRESTO associate ontologically classified concepts. Further, DICE extends JACK with meta-models (of roles, behavioural models, and other behavioural-controlling rules), provides its own high-level interpreted language (to support end-user development of custom behaviours), and allows goals to be pushed from the outside.

C. *DICE scheduler*

DICE takes complete control of JACK agents with a scheduler implementing a cyclic process that guarantees the synchronization of beliefs and intentions of the agent with the VR. This cycle is implemented as follows: first, it processes the perceptions coming from the VR in a step called SA Drilling Down; second, it generates intentions in reactions to perception; third, it gives control to the current main intention (plan) that runs until it decides to yield. During the period in which the plan runs (called "cognitive step") incoming perceptions are queued but not processed by DICE.

D. *Introspection and meta-level reasoning*

In agent oriented architectures, introspection refers to the ability of an agent to reflect upon the workings of his own cognitive functions [4]. DICE supports introspection over both goals and intentions, so agents are able to access to their own stacks of running plans and events. This allows, for instance, to pick a plan (intention) or an event (goal) from the stack and terminate and fail it.

The actions of most Artificial Intelligence (AI) programs can be classified in base-level actions, those that achieve the program's goals, and meta-level actions, involved in deciding which base-level actions to perform [5]. Meta-level reasoning in JACK is limited to choosing, from a pool of applicable plans (intentions), the best one to be executed to achieve a goal. DICE supports much more advanced meta-level reasoning thanks to its introspection facilities. As discussed later, our work exploits them in particular during the SA step of the DICE scheduler, which allows to control the inner workings of the agent at well-known times and in coordination with perceptions from the environment.

III. Architectural approach

Implicit Negotiation Coordinating Agents (INCA) is the approach that we propose here to better approximate human-like behaviors in social situations. It is a distributed and implicit approach to coordination in MASs.

The situations taken into consideration by INCA are related to access to resources, where social norms allows to implicitly negotiate the order of access. INCA extends BDI agents allowing them to integrate social norms.

A. Problem definition and principles

The question to which INCA tries to answer is: "what actions, and in which order, should intelligent actors perform, to coordinate for an ordered access to a resource?". *Actors* here are meant as rational participants in a coordination process and have the task to solve the coordination problem and perform coherent actions.

INCA relies on the key concept of *resources* that can be engaged by actors, but only by a limited number at the time. To this end, actors must be ordered in logical access *queues*. Since INCA focuses on coordination by implicit negotiation, queues are autonomously generated by each actor. If individually-generated queues are coherent one to another, thanks to pre-shared social norms, and if everybody acts respecting its own queue, the actions performed should result coordinated. Note that agents are autonomous and can therefore decide to break rules; for example, if an agent realizes to be late it could decide to not respect its turn and overcome others actors in the queues, resulting in a non coordinated behavior.

Actions are the possible interactions of the actors with the environment, and we are interested in recognizing and giving semantics to those related to coordination processes. Within INCA we have identified four actions that actors necessitate and require to perform to coordinate and access resources. These are approaching, waiting, engaging and engaged. For example, actors that want to enter a room and need to pass trough a door, first approach a queue for that door, then wait in queue, later they walk toward it and, at the end, they cross the door.

Agents reason on the information they have about the coordination context and try to filter alternative solutions, which represent possible order of access of the participants (in the simplest case a single one) by applying context-specific social norms. To simulate human-like coordination mechanisms, information about the context is constrained to what is perceived by the agent controlled character and also includes beliefs about what other actors are doing.

B. Architecture

Figure 2 illustrates the anatomy of INCA within an autonomous agent, thus repeated for each component of a MAS. It is composed of the PRESTO framework, the DLV [7] reasoner and the social norms and agent integration core modules.

PRESTO. The simulation framework on which INCA has been developed is represented at the bottom of Figure 2. As discussed earlier, PRESTO interfaces a VR with NPC controlled by autonomous agents. **DICE** is a MAS based on BDI agents adapted to reflect the typical cycle of execution of a VR. **Unity** is a complete development framework for VRs that provides the environment where NPCs live and act. PRESTO integrate them throughout the generation of perceptions and let agents interact with the VR.

DLV. The *DLV* reasoner, represented at the top of Figure 2, is exploited by INCA to perform Knowledge Representation and Reasoning (KR&R) about the coordination problem.

The social norms module. Concepts and components of the social norms module are represented in the upper part of the circle at the center of Figure 2.

Agent integration module. The agent integration module is represented in the lower part of the circle at the center of Figure 2. Its components integrate the social norms into an autonomous agent.

C. The social norms module and DLV

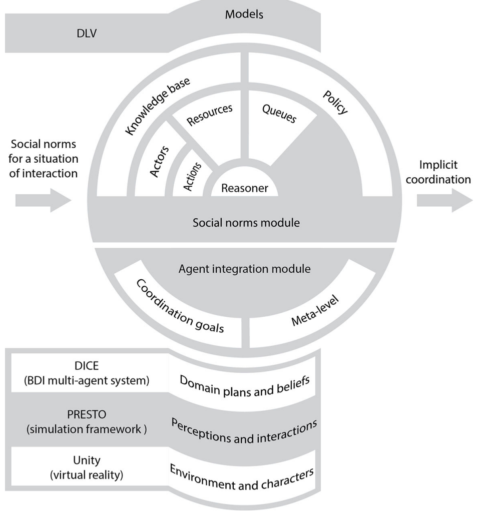
Everything related to the coordination context is represented in a Knowledge Base (KB) on which meta-level defined social norms are applied by a reasoner to resolve the coordination problem.

The representation of social norms with a meta-level language allows them to be implementation-independent with respect to agents. It is also used to represent the coordination context.

The meta-language is based on DLV, a disjunctive logic language that allows KR&R. It is currently the best implementation of a Disjunctive Logic Programming (DLP) language [6], which is the most flexible KR&R formalism. In DLP languages, facts describe an initial situations on which the application of disjunctive rules generates different solutions represented by models. DLV proposes an his own dialect of a generic DLP language, that allows additional reasoning features.

[7]Dlvsystem.com, (2016). DLVSYSTEM S.r.l. [Accessed 25 Feb. 2016]

Fig. 2. INCA anatomy



DLV allows to represent the coordination context and the social norms and also to reason on them. The coordination reasoner takes in **input** a KB about the context of coordination, which could be partial because of the limited information available to the agent. It contains information about the **actors** and the **resources**. A **queue engine** computes all possible combinations of actors in queues and generates a model for each combination. At this point, an **instructions engine** computes the instruction that each actors should follow. Then a **policy**, which is a composition of social norms valid for a specific domain of application, is applied to filters the solutions, at best, up to one.

Social norms are filtering criteria of the different solutions proposed by the queue engine. They are represented, in DLV, as weak constraint rules that rate solutions giving a penalty to the ones that does not respect them. DLV returns the solution that minimize the number of penalties.

In the example of individuals that want to pass through a door to enter a room, social norms describe commonly adopted rules to define the order of access; for instance, in normal conditions in a Western country well-behaving adults respect their order of arrival. In DLV this can be obtained with a weak constraint rule that gives a penalty to each solution where a person X, who has arrived after Y, is in the queue before the latter: **:∼ next(X,Y), queue(X,C), queue(Y,C), waiting-before-than-for-queue(Y,X,C).** As another example, consider the case of multiple doors, each with its own queue. People tend to spread so to minimize the time of access. In DLV this can be represented with a weak constraint rule that gives higher penalties to solutions where more people wait in the same queue: **:∼ next(X, Y).** The ”next(X,Y)” fact encodes that Y will access the resource after X; for each person in a queue, ”next” facts are generated to define a person's relation with people already in a queue. A long queue is a valid candidate that generates a penalty. Assume that there two persons and two doors: a solution that assigns each person to a door does not receive any penalty while a solution that queues both to a single door gets a penalty of one.

D. Agent integration module

The agent integration module integrates social norms into an autonomous agent. It transforms simple BDI agents into normative BDI agents. The *meta-level plan* is the software component that performs the coordination reasoning. It allows to keep the coordination problem separated from the domain of application. Other components are the *action execution plans*, which provide domain-specific execution capabilities to generic agent actions.

1) Meta-level: It is the component that executes the DLV reasoner. It does not make the agent to execute any action but operates only on agent's internal conditions controlling the intention deliberation, which is done thanks to introspection on the stack of active goals and plans. In particular it control the coordination goals given on the basis of the instructions computed by the reasoner.

2) Coordination goals: A coordination goal represents the intention to access some resource by coordinating with other agents. It can be achieved with specific plans, which perform coordination actions for a specific domain of application. These plans do not implement any reasoning about the coordination, which is a task delegated to the meta-level. In the example of passage through a door the actors, interpreted by agents, performs different actions; for instance, approaching a queue for a group of doors connecting a room with another one, forming and maintaining a precise disposition inside the queue, going toward a specific door and finally pass through it. Each of these coordination actions has its own implementation in an action execution plan, specific for the doors domain.

The default policy to be adopted for a given context is chosen by an agent in a domain specific plan, but the agent cognitive model can affect INCA by replacing that policy with one that reflects the current emotional state. For example, if a person is panicking and thus acting irrationally, she does not care to adopt or follow a socially accepted behavior and simply puts herself at the top of the queue.

IV. Toolset support for INCA

INCA has been integrated in PRESTO where it has been adopted to coordinate agents that need to pass through doors and gates of various types and nature. The INCA toolset provides, to PRESTO agents programmers, facilities to integrate implicit coordination capability into DICE agents.

PRESTO allows to build agents by compositions of roles, defined as sets of goals, and behavioral models, which encapsulate the implementation of these goals and are packaged as JACK modules called “capabilities”. PRESTO agent programmers develop behavioral models and all their elements and assemble roles to create agents.

The rest of this section first illustrates some guidelines for a PRESTO agent programmer on how to exploit INCA and then it illustrates its components.

A. *How to initiate the coordination*

To start to coordinate with other agents, INCA has to be started by a domain specific plan that necessitate it. These plans are developed by agent programmers and implement specific agent capabilities, e.g., walking, driving, providing medical care, etc. To start INCA from a domain plan, it is necessary to submit a domain-specific coordination goal which, in turn, causes the meta-level to start its reasoning on the coordination problem to be handled and takes control of the domain goal itself.

For example, a domain plan that handles a person navigation capability could require coordination to pass through doors. To achieve this it submits a specific INCA-enabled coordination goal. In its current implementation as part of the standard navigation capability, doors are automatically recognized when perceived and, when there is the need to cross them, INCA is started.

B. *Implementing a domain-specific coordination goal*

A coordination goal, which represents the intention to access some resource by coordinating with other agents, takes in input the policy to adopt and the resources to which the agent is interested in. Its execution automatically triggers the meta-level of INCA, which, in turn, starts to reason on the coordination problem and the invocation of domain-specific plans implementing coordination actions. Our current implementation includes a goal to pass through doors and related plans. The "approaching" plan moves the character toward the last person in the queue, a "waiting" plan forces to stay behind the last person in the queue, the "engaging" plans makes the character walk toward the door. To allow the automatic recognition of actions performed by other NPC participating to the coordination, we exploit the ability of PRESTO of ontologically tagging entities in the VR to expose the current action execution plan. Alternative implementations, e.g. to manage crowding in front of doors rather than ordered queuing, will be created in future.

C. *Integration of INCA in PRESTO - the meta-level*

The core of the integration of INCA in PRESTO consists in meta-level facilities where social norms are added to the agent architecture, allowing to separate coordination reasoning from actions execution. It exploits the SA step of the DICE scheduler to handle perception updates concerning NPC participating to the same coordination; this involves generating a KB and starting and monitoring an external process executing DLV to compute the resource-access queue and decide which action to take next. DICE introspection is called to identify active coordination goals and change the current action execution plan when required.

D. *Evaluation - scalability test*

In this section is presented a scalability test of the coordination reasoner, which is the active component of INCA.

Fig. 3. Plot per people

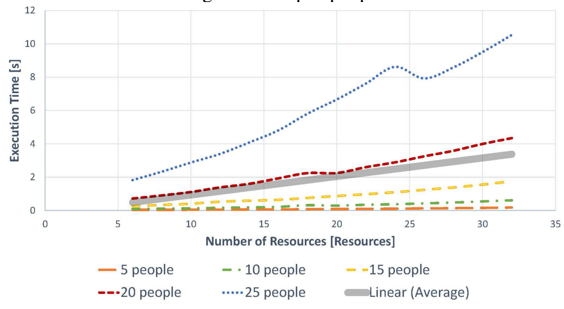
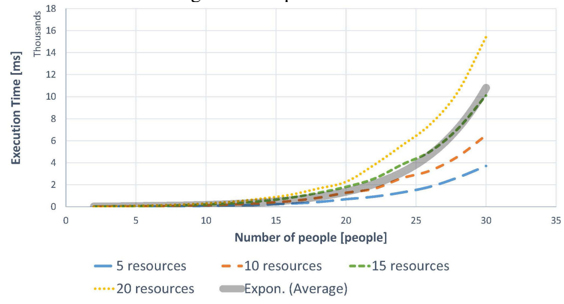


Fig. 4. Plot per resources



1) Design of the experiment: We evaluated the scalability of the reasoner with respect to the execution time over factors of complexity. Two of them have been taken into consideration: the number of people and the number of resources. The coordination reasoner takes in input a policy, a very simple one has been adopted, and a KB representing the coordination context.

2) Method of execution: The tests have been ran on a machine, with an Intel(R) Core(TM) i7-5500U @ 2.4Ghz and 8GB of RAM. The operating system is Windows 10 by Microsoft(R). A Java program has been developed to sistematically generate the KB, run the DLV process and measure its execution time. It runs the DLV process 5 times, for each combination of the factors of complexity, and measures the execution times. From the 5 measures, of the same combination, the higher and the lower are removed, then the remaining 3 are averaged.

3) Results interpretation and conclusion: Figure 3 shows the execution time in variations to the number of resources. It shows five series of tests executed with different numbers of people: 5, 10, 15, 20, 25. The tendency seems to be linear or polynomial with respect to the number of resources. Figure 4 shows the execution time in variations to the number of people. It shows four series of tests executed with different numbers of resources: 5, 10, 15 and 20. The tendency of all the four series seems to be exponential with respect to the number of people.

The execution time of a single run of the reasoner increases with the increment of both the number of people or the number of resources, this is due to the increasing number of possible solutions, each bounded to a combination of the actors in the queues. It is important to consider that the KB have been systematically generated and, in a real case, more constraints could simplify the problem, while a complex policy could complicate it. Consider also that, in a real case, each agent executes an instance of the reasoner by his own. INCA approach has been developed for VR systems, where real-time is an important aspect, and also the performances of the coordination reasoner could contribute in obtaining a realistic simulation.

V. Related work

This section presents some research works about multi-agent based simulations of societies of people and the problem of coordination in MASs.

1) Social norms and autonomous agents: Shoham in [7] defines a social norm as a restriction of the set of actions available to the agents.

2) Normative agents: Deliberative normative agents are agents that have an explicit knowledge about the enacted norms in a multi-agent environment and can make a choice whether to obey the norms or not [8]. Luck et al. in [9] study how norms can be incorporated into autonomous agents. They propose some strategies to constrain agent autonomy through norms and define a process of autonomous norms compliance, which is a different thing to a simple adoption of norms.

3) Coordination approaches in MAS: For Wooldridge in [10] the coordination problem is that of managing inter-dependencies between the activities of agents. A basic approach to coordinate multiple agents is to restrict their activities in a way which enables them to achieve their goals while not interfering with other agents [11].

4) Negotiation: Coordination can adopt implicit negotiation, where agents do not explicitly communicate, but negotiation is embedded in a pre-existing context [12]. In Michael et al. [13] is studied the problem of interaction between intelligent agents without the benefit of communication. They examine various constraints on the actions of agents and discuss their effects.

5) Coordination by norms and social laws: Savarimuthu et al. in [14] show that norms facilitate coordination and cooperation among the members of a society. Jennings in [15], after having studied the theoretical problem of coordination, concludes with the need to identify actions and social norms (he called them commitments and conventions) on which agents are able to performs reasoning. Shoham et al. in [16] ask themselves why not adopt a convention, or, as we would like to think of it, a social law, according to which if each agent obeys the convention, there will be avoided a lot of interactions, creating an implicitly coordinated social behavior without any need for either a central arbiter (to be avoided in MAS) or negotiation (also complex in MAS).

VI. Conclusion

We propose INCA, an implicit negotiation approach for a multi-agent simulation of human-like coordination mechanisms based on social norms. INCA provides a meta-level language and reasoner, integrated in the architecture of a BDI agent. These components allow agents to reason on and follow social norms about queues of access to resources. INCA uses MAS technology to mimic an implicit negotiation of these queues, allowing agents to coordinate with others for the usage of resources. INCA can be therefore exploited to simulate implicit coordination mechanisms. The INCA toolset has been implemented in PRESTO and in order to evaluate its performance in complex scenarios, we performed scalability tests. Future work include a more efficient and usable implementation of INCA.

References

[1] M. Dragoni, C. Ghidini, P. Busetta, M. Fruet, and M. Pedrotti, "Using Ontologies For Modeling Virtual Reality Scenarios," in *Proceedings of ESWC 2015*, 2015.

[2] P. Busetta and M. Dragoni, "Composing Cognitive Agents from Behavioural Models in PRESTO," in *Proceedings of the 16th Workshop "From Objects to Agents" (WOA-2015)*, 2015.

[3] P. Busetta, R. Rönnquist, A. Hodgson, and A. Lucas, "Jack intelligent agents-components for intelligent agents in java," *AgentLink News Letter*, vol. 2, no. 1, pp. 2–5, 1999.

[4] K. Konolige, "A Computational Theory of Belief Introspection," in *IJCAI*, vol. 85, 1985, pp. 503–508.

[5] M. Genesereth, "An Overview of Meta-Level Architecture," in *AAAI-83 Proceedings*, 1983, pp. 119–124.

[6] N. Leone, G. Pfeifer, W. Faber, T. Eiter, G. Gottlob, S. Perri, and F. Scarcello, "The DLV system for knowledge representation and reasoning," *ACM Trans. Comput. Logic*, vol. 7, no. 3, pp. 499–562, 2006.

[7] Y. Shoham and M. Tennenholtz, "On the emergence of social conventions: modeling, analysis, and simulations," *Artificial Intelligence*, vol. 94, no. 1-2, pp. 139–166, jul 1997.

[8] C. Castelfranchi, F. Dignum, C. M. Jonker, and J. Treur, "Deliberate Normative Agents: Principles and Architecture," *Intelligent Agents VI*, vol. LNAI 1757, pp. 364–378, 2000.

[9] M. Luck and M. D'Inverno, "Constraining autonomy through norms," in *Proceedings of the First International Joint Conference on Autonomous Agents and Multiagent Systems part 2 (AAMAS-02)*, 2002, pp. 674–681.

[10] M. Wooldridge, *Introduction to MultiAgent Systems*. Hoboken, NJ, USA: Wiley, 2002.

[11] Y. Shoham and M. Tennenholtz, "On social laws for artificial agent societies: off-line design," *Artificial Intelligence*, vol. 73, pp. 231–252, 1995.

[12] F. Scharpf, "Games Real Actors Could Play: Positive and Negative Coordination in Embedded Negotiations," *Journal of Theoretical Politics*, vol. 6, no. 1, pp. 27–53, 1994.

[13] M. Genesereth, M. Ginsberg, and J. Rosenschein, "Cooperation Without Communications," in *AAAI-86 Proceedings*, Stanford Heuristic Programming Project, Computer Science Department, Stanford University. Heuristic Programming Project, Computer Science Department, Stanford University, 1984.

[14] T. Savarimuthu, M. Purvis, and M. Purvis, "Social norm emergence in virtual agent societies," in *Declarative Agent Languages and Technologies*. Berlin: Springer, 2008, vol. VI, pp. 18–28.

[15] N. Jennings, "Commitments and Conventions: The Foundation of Coordination in Multi-Agent Systems," *The Knowledge Engineering Review*, vol. 8, no. 3, pp. 223–250, 1993.

[16] Y. Shoham and M. Tennenholtz, "On the synthesis of useful social laws for artificial agent societies," in *AAAI-92 Proceedings*, 1992.