Computational Institutions for modelling Norm-Regulated MAS: an approach based on coordination artifacts

Rossella Rubino[1] *, Andrea Omicini[2], and Enrico Denti[3]

[1] CIRSFID, Alma Mater Studiorum – Università di Bologna
Via Galliera 3 – I-40121 Bologna, Italy
[2] DEIS, Alma Mater Studiorum – Università di Bologna a Cesena
Via Venezia 52 – I-47023 Cesena, Italy
[3] DEIS, Alma Mater Studiorum – Università di Bologna
Viale Risorgimento 2 – I-40136 Bologna, Italy

Abstract As agent autonomy emphasises the need of norms for governing agent interactions, increasing attention is being devoted to (electronic) institutions for modelling organisations governed by norms. Moving from the concepts of role (with its normative consequences, i.e. obligations, permissions and prohibitions), norms (both regulative and constitutive), and normative agents, we first introduce the notion of Computational Institution for modelling norm-regulated MAS. Then, we discuss how infrastructural abstractions like coordination artifacts can be exploited to express norms inside computational institutions, and present an example based on the TuCSoN infrastructure.

1 Norms and Agents

Generally speaking, *norms* are rules, enforced by some (trusted) third parties, aimed at governing the individual behaviour of the members of a society. Although the most common semantics of norm recalls the idea of *imposing* a specific rule or behaviour, this is not always the case in real life. According to Searle's classification [1], norms can be classified in two categories:

- *constitutive norms*, i.e. norms that are affirmed to create (constitute) new states of affairs (example: the rules of a game, like chess);
- *regulative norms*, also called *deontic rules* [2], i.e. norms that are aimed at governing activities, by expressing the obligation or the permission to perform an action (example: "you should drive on the right").

According to Peczenik [3], a special case of constitutive norms is the case of *qualification norms*, which are defined as constituting some particular legal properties; among these, notable examples are *norms that confer competence* and *norms that*

* Corresponding author. Email: rrubino@cirsfid.unibo.it

confer power — i.e, norms that constitute a specific ability for some specific entity. On the other hand, regulative norms are further classified into *behavioural norms* and *aim norms*.

In the context of MAS, we will mainly refer to Searle's classification, since the basic distinction is between norms for ruling social activities (which fall inside the class of regulative norms), and norms for ascribing responsibilities and creating new concepts (which belong to the class of constitutive norms). In particular, regulative norms should both enable each agent to achieve its goal(s), and allow interactive social activities to be handled as virtual organisations or societies, somehow mediating among different exigencies.

In this paper, we introduce the concept of *computational institution* as a model to formally include the notion of norm into virtual organisations, and show how computational institutions can be actually represented and effectively set up via suitable *coordination artifacts* [4] exploited as *normative abstractions*. As a concrete example, we outline a case study showing how to build a simple computational institution on top of the TuCSoN coordination infrastructure.

2 Organisations and Virtual Institutions

From an abstract viewpoint, an organisation can be defined as "a social unit or human grouping deliberately constructed to seek specific goals" [5]. Institutions, in their turn, can be introduced as "the framework within which interaction takes place" [6]: they provide a society with the structure and the rules (constraints) needed to shape interaction among its participants. Among the models defined in the literature to frame the concept of virtual institution, Noriega and Sierra introduced the notion of *electronic institution* [7,8], later extended by Vasconcelos by introducing the notion of *logic-based e-institution* [7]; on the other hand, in the MAS context, Boella and Van der Torre defined the notion of *normative system* [9,10,11] as a MAS with norms. Altogether, these models introduce the key concepts of *deliberative agent*, *role*, *norm*, and *normative agent*.

Both electronic institutions and normative systems adopt a notion of agent which emphasises agent autonomy: their agents can decide to violate a norm to achieve their goals, or to change their goals so that they match the existing norms — a property called *norm autonomy*. This is why such agents are known as *norm autonomous agents* or *deliberative normative agents* [12] — in the following, just *deliberative agents*, for short.

Each agent in the institution can play one or more *roles*, which determine what an agent can do: basically, the concept of role is common to both approaches. Roles may be shared by several agents, and may be acquired either statically or dynamically.

On the contrary, *norms* are not seen in the same way in the approaches above, since normative systems feature both regulative and constitutive norms, while electronic institutions consider regulative norms only. So, the electronic institution norms can be seen as a subset of the normative system norms. In particular, electronic institutions define regulative norms that may be submitted

to preconditions — a natural choice, since they mainly focus on communication languages and interaction protocols.

The concept of *normative agent* — i.e., a member of the institution whose goal is to enforce norms —, instead, is unique to normative systems.

3 Computational Institutions

A *Computational Institution* is a virtual organisation ruled by norms intended as in Section 1. The word "computational" means that the entities participating in the institution are not only humans, but also computational virtual entities that operate in order to achieve the social shared goal(s). As in real life, the main institutional tasks [8] are:

- to manage the identity of the participants;
- to define and validate the requirements on participant capabilities;
- to establish interaction conventions;
- to enforce the possible obligations.

With respect to the first issue, each member of the institution is characterised by its identity, and by the role played in the institution. Of course, participant identity management is essential, for both social and legal reasons: on the one hand, knowing participants' identities might be necessary to perform some tasks, or just to facilitate their collaboration; on the other, knowing their identities also makes it easier for the normative agents to perform their tasks.

As regards the second issue, every agent works in the institution in order to achieve individual/social goal(s) by playing one or more roles which describe what actions it can do. Each role can be associated to some requirements that the agents playing such role(s) should fulfill in order to be a member of the institution.

Computational institutions consider three main roles:

- the *legislative* role, which consists of making laws;
- the *judicial* role, which consists of deciding whether there is a violation;
- the *executive* role, which consists of detecting violations and enforcing norms by applying the proper sanctions.

So, unlike the institutions considered in the previous Section, computational institutions consider three normative agents — one for each of the above roles; of course, in order to ensure objectivity, an agent can be assigned only one of them (as in any "well formed" political structure).

More formally, a computational institution can be defined by the n-ple:

$$\langle A, R, Req, N, G, aL, aJ, aE, S, Act \rangle$$

where

- A is the set of the agents participating to the institution;

- R is the set of the roles that agents can play;
- Req is the set of the requirements that each agent should satisfy to be a member of the institution;
- N is the set of the norms ruling institution execution;
- G is the institution goal (shared between all institution members);
- aL is the legislative agent;
- aJ is the judicial agent;
- aE is the executive agent;
- S is the set of the sanctions;
- Act is the set of the activities that need to be performed for the institution goal to be achieved.

As a simple modelling example, let us consider the case of *virtual enterprises* (VE henceforth) [13]. A VE is a temporary aggregation of autonomous and usually heterogeneous enterprises, aimed at achieving a common goal, and whose light-weight structure is well-suited to face the frequent changes and openness of business scenarios in a flexible and adaptable way. As an aggregation of enterprises, a VE requires suitable norms and sanctions, possibly negotiated in its set-up phase, so as to govern the mutual dependencies and coordinate the individual enterprises, as well as the definition of the roles needed to represent the tasks to be performed in order to achieve the VE's goal. So, a VE can be interpreted as a computational institution, where:

- A is the set of agents which cooperate in the VE;
- R is the set of roles that VE agents can play — that is, client, VE initiator, and VE partner;
- Req is the set of the requirements that each VE agent must satisfy in order to be admitted to participate to the VE;
- N is the set of the norms ruling the VE, established either in the (initial) negotiation phase, or by some other higher-level entity, such as the Government, State, etc.;
- G is the VE goal;
- aL is the VE initiator or client;
- aJ and aE are the competent bodies;
- S is the set of sanctions, again (like norms above) either negotiated or established by some other entity;
- Act is the set of activities needed to achieve the VE goal.

A more detailed example will be discussed in Section 5.
Let us sum up the advantages and disadvantages of computational institution with respect to electronic institutions and normative systems. Among the advantages, there is both the chance to define the requirements that agents should fulfill in order to be members of the institution, and the separation of powers (executive, judicial and legislative), which enables to distinguish the different roles. Moreover, by abstracting from technical details, computational institutions can be seen as a general framework for virtual institutions. Despite these advantages, computational institutions do not consider agent communication, nor do they

make it possible to describe a simple structure where one agent plays the three roles (executive, judicial and legislative) at the same time.

So far, we discussed the definition of computational institution and its composing elements based on a legal analysis of human institutions. With respect to the social aspects of computational institutions, we should examine the activities performed by agents in order to achieve their goals.

According to the research studies in the field of human (cooperative) activities, mainly in Activity Theory [14,15], non-trivial human activities are always mediated by some kind of artifacts, that enable and mediate interaction, ruling/governing the resulting global and "social" behaviour [16]. In fact, artifacts are widespread in human society: the language can be considered an artifact, as well as the writing, blackboards, maps, post-its, traffic signs such as semaphores, electoral cards or the signature on a document.

Based on this background, *coordination artifacts* were recently introduced as a conceptual and engineering framework for MAS and agent societies [4,16]: our goal is to exploit coordination artifacts also for the engineering of computational institutions in MAS. So, in next Sections we explore this aspect, discussing in particular how two infrastructural abstractions — namely, coordination artifacts and agent coordination contexts — can be exploited for modelling and engineering computational institutions as MAS.

4 Coordination Artifacts for Computational Institutions

A coordination artifact [4,16] is a conceptual and run-time abstraction aimed at entailing a form of mediation among the agents that use it, and embedding and effectively enacting some coordination policy (i.e. suitable laws and norms). Accordingly, from our viewpoint, coordination artifacts feature both a *constructive* and a *normative* nature, as they take care respectively of creating/composing social activities, and of ruling/governing them.

So, in order to map the model of computational institution onto the notion of coordination artifact, it is convenient to conceptually separate the computational institution items into two corresponding subsets:

- *constructive items*, i.e. the sets of agents (A), roles (R), requirements (Req), and activities (Act), as well as the institution goal (G);
- *normative items*, i.e. the sets of norms (N), sanctions (S), and the normative agents (aL, aJ, aE).

Of course, this not means that coordination artifact encapsulate all such aspects.

From a "norm-oriented" viewpoint, the *encapsulation*, *malleability*, *inspectability* and *controllability* properties of coordination artifacts are particularly relevant, since they correspond to desired properties of computational institutions.

Encapsulation means that a coordination artifact encapsulates a coordination service, allowing user agents to abstract from the actual service implementation. In the context of computational institutions, this translates in the chance for user agents to abstract from how normative agents actually rule/supervise the

institutional activity (and possibly punish any "illegal" behaviour). Malleability, in its turn, represents the ability of a coordination artifact to be adapted and changed dynamically, following the intrinsic dynamism and unpredictability of MAS — an aspect which is common also to computational institutions — so that agents can enter/exit the system, or change their role, at any time. Analogously, inspectability/controllability of the coordination artifact structure enable agents to use/control the artifact correctly; indeed, a worthy feature of computational institution is precisely the inspectability of its dynamic state, i.e., of what happens during the institution run-time. This property makes it possible to inspect/access any stored information about interaction histories and events occurred within an institution, for instance for normative purposes like incorrect behaviour detection and violation detection.

TuCSoN [17] is an example of agent coordination infrastructure supporting a notion of coordination artifact called *tuple centre* [18]. Tuple centres are programmable tuple spaces — i.e., sort of reactive, logic-based blackboards — that agents access by writing, reading, and consuming tuples — that is, ordered collections of heterogeneous information chunks — via simple communication operations (*out*, *rd*, *in*), which access tuples associatively. While the behaviour a tuple space in response to communication events is fixed, the behaviour of a tuple centre can be programmed by defining a set of specification tuples expressed in the ReSpecT language [18], which define how a tuple centre should react to incoming/outgoing communication events. As a result, tuple centres can be seen as general-purpose customisable coordination artifacts, whose behaviour can be dynamically specified, forged and adapted so as to automate the co-ordination stage among agents [16].

Topologically, tuple centres are collected in TuCSoN coordination nodes, spread over the network: each node constitutes an organisation context. In order to access/use the tuple centres of an organisation context, an agent must first negotiate and enter an *Agent Coordination Context* (ACC henceforth, [19]), which defines the agents presence and position inside the organisation in terms of actions allowed on tuple centres by virtue of the agent's role(s). More precisely, an ACC is meant both to model of the environment where the agent interacts, and to enable/rule the interactions between the agent and the environment, by defining the space of the admissible agent interaction.

As discussed above, the set of norms comprises regulative and constitutive norms. Since regulative norms aim at governing activities by expressing the obligation, the permission, or the prohibition to perform an action, they can be naturally managed by agent coordination contexts. On the other hand, constitutive norms may be embedded into coordination artifacts, e.g. by defining a suitable tuple or by programming the tuple centre so as to react to selected events. The corresponding specification tuples could be inserted in the tuple centres by the legislative agent.

Accordingly, while tuple centres can be used to model the social aspect of computational institutions, embedding the corresponding norms, ACCs can be exploited to model the presence of an agent in a computational institution with

respect to organisation, access control, and relationships between agents and institution. So:

- in order to be member of a computational institution, an agent must first obtain an ACC, which defines its role in the institution;
- the ACC can also be interpreted/exploited as a legal artifact defining what kinds of interaction service(s) were promised to the agent by the infrastructure — and, conversely, what kinds of actions the agent can be expected to execute given his role(s).

Summing up, coordination artifacts are suitable tools for building computational institutions, possibly in conjunction with other abstractions — such as ACCs — for the management of roles, requirements and regulative norms.

Next section discusses a simple example, showing how to define a computational institution on top the TuCSoNinfrastructure artifacts.

5 Example: public competitive tender

In this Section, we show that a simple *public competitive tender* may be seen as a computational institution. Public competitive tenders are onerous contracts stipulated between one o more economic operators, called *bidders*, and one or more *awarding administrations*, whose subject is the execution of a work or the supply of a product or service [20]. Among the various kinds of procedures usually adopted for competitive tenders, we consider here only the so-called "open procedure", i.e., the procedure adopted when any economic operator can participate to the tender as a bidder.

In the first step, the awarding administration publishes the announcement of the competitive tender; then, the examining commission is set up, according to the criteria defined in the announcement, which also states the acceptance criteria for the bidders' offers. Usually, the most common criteria for acceptance are either the *most advantageous offer* (evaluating altogether quality, price, feature, usage cost, terms of delivery, etc.) or simply the *lowest price*. When all the (valid) bids have been examined, the awarding administration eventually announces the winner.

Public competitive tenders can be represented as computational institutions according to the definition introduced in Section 3, where:

- A is the set of the agents involved in competitive tenders;
- R is the set of roles occurring in a competitive tender as defined above, that is the awarding administration, the economic operators, the members of the examining commission;
- Req is the set of the requirements stated in the tender announcement, concerning skills and abilities required from the economic operators, such as, for instance, economic skills, financial abilities, technical or professional skills, etc.;
- N is the set of all the norms ruling the competitive tender according to the laws/directives in force (such as [20]);

- G is the institution goal, which in this case is to stipulate a contract between the awarding administration and the winner economic operator;
- aL is the legislative agent which issue the law/directive (e.g. the European Union [20]);
- aJ is the judicial agent (e.g. judges, tribunals, etc.);
- aE is the executive agent, charged of enforcing the norms by applying sanctions (e.g. the police or some public officer);
- S is the set of the sanctions, usually listed in the norms stated by the legislative agent, to be applied in case of violations — for instance, exclusion from the current competitive tender, and possibly from further tenders;
- Act is the set of activities to be performed inside the procedure, such as emitting the announcement, setting up the examining commission, evaluating the offers, etc.

Mapping such a computational institution onto TuCSoN coordination artifacts amounts at adopting tuple centres to capture the institution's social aspects, and ACCs to model agent individual issues with respect to the environment. So, a tuple centre could be charged of governing interaction among the tender's agents, according to the tender's procedure; moreover, suitable ACCs should be introduced for each of the different tender's roles/agents. Then, a suitable behaviour specification should be defined in order to enforce the norms related to agent interaction, and possibly also some norms related to regulative aspects; however, how much of the burden of norm enforcing should be charged onto specific normative agent(s), and how much should be embedded into the programmable coordination artifacts, is an open design dimension.

In particular, the "open procedure" described above may be expressed by introducing three agent roles (the awarding administration, *AWA*; the member of examining commission, *MEC*; and the bidder, *B*) along with the related interaction protocols in terms of exchanged tuples, and by representing the procedure phases, too, as suitable tuples.

Figure 1 shows a competitive tender in which the administration `name` publishes the announcement, and two bidder agents, *B1* and *B2*, participate to the tender. Of course, some commission will decide the winner. So, suitable ACCs should be introduced for each of the different tender's roles/agents: the awarding administration, the member of examining commission, and the bidder, *B*. Then, for instance, the procedure could take place as follows:

1. the agent playing the role of the awarding administration, *AWA*, publishes the announcement by inserting the `announcement/4` tuple :

   `announcement(idA(`*`id`*`),admin(`*`admin`*`),subject(`*`s`*`),criterion(`*`c`*`))`

   where *`id`* is the announcement's unique identifier, *`admin`* is the awarding administration name, *`s`* is the subject of the contract, and *`c`* is the criterion (most advantageous offer or lowest price) to be used for this tender. In response, the tuple centre's supposed behaviour is to trigger the automatic

insertion of a `commission_member/2` tuple for each required member of the examining commission, i.e. for each *MEC* agent[4]:

`commission_member(idA(`*id*`),admin(`*admin*`))`

2. the agents playing as bidders express their intention to bid for this tender by inserting an `offer/3` tuple such as

   `offer(idA(`*id*`),bidder(`*bidder*`),price(`*price*`))`

   where *bidder* is, rather obviously, the bidder agent's identifier, and *price* is the offered price.
3. when the offer deadline expires, *MEC* agents (properly coordinating themselves via some interaction protocol enforced by the tuple centre rules) gather all the offers, and select the winner according to the criterion specified in the announcement. As a result, a `winner/2` tuple is emitted to publish the winner's name (see Figure 1).

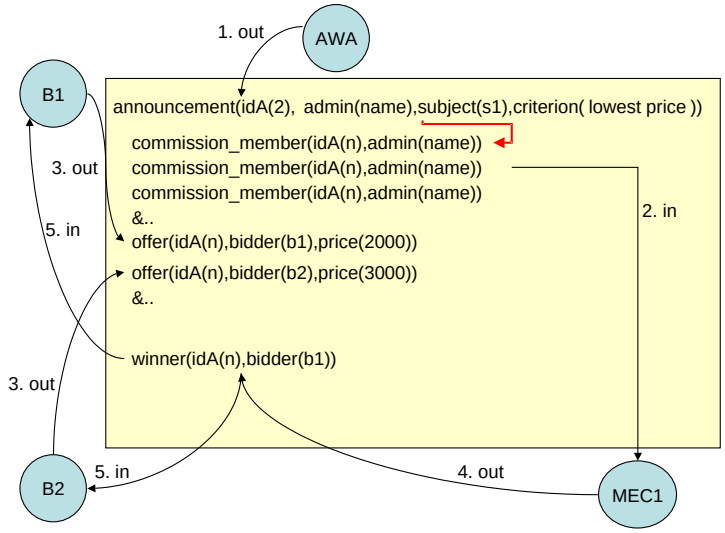

Figure 1. Mapping a computational institution onto TuCSoN: the tender example.

Constitutive rules may define the tuple templates used in the institution, as follows:

- `const_norm(announcement(idA,admin,subject,criterion('lowest price', 'more advantageous offer')))`
- `const_norm(commission_member(idA,admin))`

[4] Their number is supposed to be a constitutive norm, expressed as a suitable tuple, too (not shown).

– `const_norm(offer(idA,bidder,price))`
– `const_norm(winner(idA,bidder))`

These tuples may be stored in another tuple centre or in the same as the tuples offer, announcement, etc.

Let us now consider a violation scenario: for example, in Figure 2, the tender criterion (published in the announcement) is the lowest price, but the commission communicates that the winner is an agent that offered more than another. The executive agent monitoring the tender should then insert a tuple `sanction/2` describing the agent involved in the violation, along with the type of sanction. The judicial agent will finally insert the tuple related to the correct winner.

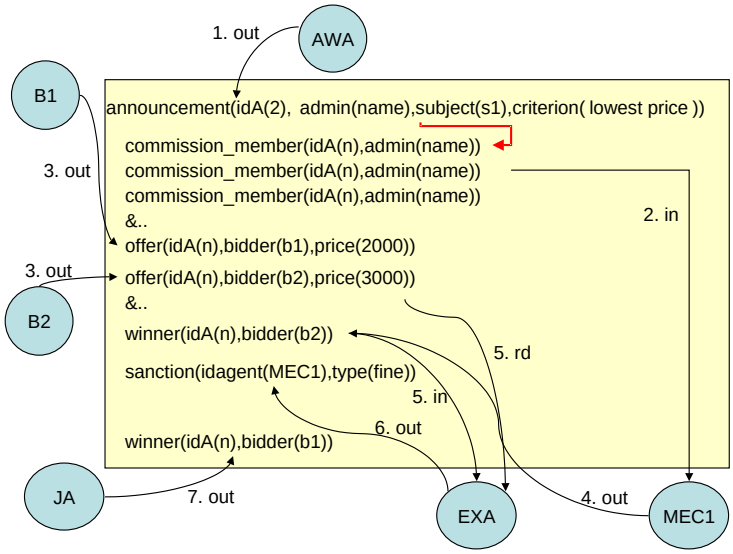

Figure 2. Violation Scenario.

6 Related work and Conclusions

In this paper we analysed the concept of norm both in the legal and coordination field. Although several analogies exist, there are different nuances of meaning: for instance, coordination policies rule only agent coordination, while norms are typical of the institutional goal, and therefore their violation entails sanctions [2].

This general concept of norm is common to all virtual institutions examined so far, that is computational institutions, electronic institutions and normative

systems. Some differences comes out on the distinction between constitutive norms and regulative norms. Indeed, while agents and roles are intended basically in the same way, normative rules in electronic institutions only include regulative norms. Constitutive norms, on the other hand, exist in Boella and Van der Torre's normative system framework. Furthermore, electronic institutions define concepts such as *dialogic framework*, *scene* and *performative structure* that are particularly suited to capture the issues related to the communication languages and protocols, and that have no counterpart in computational institutions.

Of course, several other test cases need to be mapped as computational institutions to validate this approach. Moreover, some key aspects deserve a deeper investigation: in particular, further work is needed to better explore the issues related to mapping computational institutions onto TuCSoN coordination artifacts, with special regard to the critical issue of suitably mapping the (different kinds of) norms.

References

1. Searle, J.R.: The construction of social reality. New York: The Free Press (1995)
2. Sartor, G.: Legal Reasoning: a Cognitive Approach to the Law. Springer (2005)
3. Peczenik, A.: On Law and Reason. Kluwer (1989)
4. Omicini, A., Ricci, A., Viroli, M., Castelfranchi, C., Tummolini, L.: Coordination artifacts: environment-based coordination for intelligent agents. In Jennings, N.R., Sierra, C., Sonenberg, L., Tambe, M., eds.: 3rd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2004), New York, USA. Volume 1., ACM Press (2004) 286–293
5. Etzioni, A.: Modern organisations. Prentice-Hall (1964)
6. Esteva, M., Rodriguez-Aguilar, J.A., Sierra, C., Garcia, P., Arcos, J.L.: On the formal specification of electronic institutions. In Dignum, F., Sierra, C., eds.: Agent Mediated Electronic Commerce: The European AgentLink Perspective. Volume 1991 of LNAI., Springer (2001) 126–147
7. Vasconcelos, W.W.: Logic-based electronic institutions. In Leite, J., Omicini, A., et al, L.S., eds.: Declarative Agent Languages and Technologies: First International Workshop (DALT 2003), Melbourne, Australia. Volume 2990 of LNCS., Springer (2003) 221–242
8. Noriega, P., Sierra, C.: Electronic institutions: Future trends and challenges. In Klusch, M., Ossowski, S., , Shehory, O., eds.: Cooperative Information Agents VI:6th International Workshop (CIA 2002), Madrid, Spain. Volume 2446 of LNAI., Springer (2002) 14
9. Broersen, J., Dastani, M., Hulstijn, J., van der Torre, L.W.N.: Goal generation in the boid architecture. Cognitive Science Quarterly **2** (2002) 428–447
10. Boella, G., van der Torre, L.W.N.: Regulative and constitutive norms in normative multiagent systems. In: 9th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2004). (2004) 255–266
11. Boella, G., van der Torre, L.W.N.: Attributing mental attitudes to normative systems. In: 2nd International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS 2003),Melbourne, Australia, ACM Press (2003) 942–943
12. Castelfranchi, C., Dignum, F., Jonker, C.M., Treur, J.: Deliberative normative agents: Principles and architecture. In Jennings, N.R., Lesperance, Y., eds.: 6th

International Workshop on Agent Theories, Architectures, and Languages (ATAL 1999), Orlando, USA. (1999) 364–378
13. Petersen, S.A., Divitini, M., Matskin, M.: An agent-based approach to modelling virtual enterprises. In: International Enterprise Modelling Conference (IEMC 1999), Verdal, Norway. (1999) 10
14. Engestrm, Y., Miettinen, R., Punamki, R.L.: Perspectives on Activity Theory. Cambridge University Press (1999)
15. Nardi, B.A.: Context and Consciousness: Activity Theory and Human-Computer Interaction. MIT Press (1996)
16. Ricci, A., Omicini, A., Denti, E.: Activity Theory as a framework for MAS coordination. In Petta, P., Tolksdorf, R., Zambonelli, F., eds.: Engineering Societies in the Agents World III. Volume 2577 of LNCS. Springer (2003) 96–110 3rd International Workshop (ESAW 2002), Madrid, Spain, 16–17 September 2002. Revised Papers.
17. Omicini, A., Zambonelli, F.: Coordination for Internet application development. Autonomous Agents and Multi-Agent Systems **2** (1999) 251–269 Special Issue: Coordination Mechanisms for Web Agents.
18. Omicini, A., Denti, E.: From tuple spaces to tuple centres. Science of Computer Programming **41** (2001) 277–294
19. Omicini, A.: Towards a notion of agent coordination context. In Marinescu, D.C., Lee, C., eds.: Process Coordination and Ubiquitous Computing. CRC Press (2002) 187–200
20. EU: Official journal. (http://europa.eu.int/eur-lex/pri/it/oj/dat/2004/l_134/l_13420040430it00010113.pdf)