ADOPT JaCaMo: Accountability-Driven Organization Programming Technique for JaCaMo

Matteo Baldoni[1], Cristina Baroglio[1], Katherine M. May[2], Roberto Micalizio[1], and Stefano Tedeschi[2]

[1] Università degli Studi di Torino — Dipartimento di Informatica
`firstname.lastname@unito.it`
[2] Università degli Studi di Torino
`firstname.lastname@edu.unito.it`

Abstract. This work concerns the challenge of computational accountability in a multiagent setting where agents interact inside organizations. We illustrate the requirements to realize accountability with the help of a scenario. Then, we provide a characterization of computational accountability in terms of a few general principles. We introduce and explain the ADOPT accountability protocol and show how it satisfies these principles with the help of model checking.

Keywords: Computational Ethics, Accountability, Multiagent Systems, Sociotechnical Systems.

1 Introduction

JaCaMo [6] is a conceptual model and programming platform that integrates agents, environments, and organizations. It is built on the top of three platforms: Jason [7] for programming agents, CArtAgO [22] for programming environments, and Moise [19] for programming organizations. The aim of the framework is both to integrate the cited platforms and to integrate the related programming meta-models to simplify the development of complex multiagent systems. The presence of an actual programming platform fills the gap between the modeling level and the implementation level.

According to [18], the Moise+ organizational model, adopted in JaCaMo, explicitly decomposes the specification of an organization into three different dimensions. The *structural* dimension specifies roles, groups, and links between roles in the organization. The *functional* dimension is composed of one (or more) scheme(s) that elicits how the global organizational goal(s) is (are) decomposed into sub-goals and how these sub-goals are grouped in coherent sets, called missions, to be distributed to the agents. Finally, the *normative* dimension binds the two previous dimensions by specifying the roles' permissions and obligations for missions. One important feature of Moise+ [19] is to avoid a direct link between roles and goals. Correspondingly, roles are linked to missions by means of permissions and obligations, which consequently maintain an independence between the functional and structural specifications.

In the field of multiagent systems, individual and organizational actions have social consequences, which require the development of tools to trace and evaluate principles' behaviors and to communicate good conduct. This concerns the value of *accountability*. The independence between roles and goals in JaCaMo creates difficulties when reasoning about accountability. Namely, problems result from a scheme's dynamic creation and group assignment that can happen when agents are already playing associated roles. This means that agents, *when entering into an organization* by adopting an organizational role, *have no information about what they could be obliged to do in the future* because this information, related to a specific scheme, may be not available or even not yet present. The aim of this paper is to present an Accountability-Driven Organization Programming Technique (ADOPT) that attempts to face the challenges of handling accountability computationally in an organization of agents. The main contribution is to provide a notion of when accountability can be ascribed in an organization by investigating the organization-construction process as well as to define a protocol that ensures the design and construction of an accountability-supporting organization. The core of the analysis is the notion of role and the action of role adoption (or enactment). With some conceptual modifications, we believe JaCaMo [6] a particularly suitable platform for building in an accountability mechanism. On the other hand, the principles by which we characterize accountability provide a general account of accountability in organizational settings. The paper begins with a scenario that explains in a practical way the lack of accountability and its origin in systems realized by means of JaCaMo. Then, it provides a characterization of computational accountability including five founding principles. It describes ADOPT, an accountability protocol, and shows how it satisfies these principles with the help of model checking.

2 Lack of Accountability: A Scenario in JaCaMo

In order to illustrate the accountability problem, we use, as a reference scenario, an excerpt of the *building-a-house* example presented in [6]. An agent, called Giacomo, wants to build a house on a plot. In order to achieve this goal, Giacomo will have to hire some specialized companies and then ensure that the contractors coordinate and execute in the right order the various tasks and subgoals. Each hired company must adopt a corresponding role in the organization. Roles are gathered in a group that is responsible for the house construction. After goal adoption, a company agent could be asked (through an obligation issued by the organization) to commit to some "missions". Now, let's suppose that Giacomo is a dishonest agent and wants to exploit the contracted companies in order to achieve some purposes that are unrelated to the house construction. In particular, let's suppose he wants to delegate a `do_a_very_strange_thing` goal to the agent playing the `plumber` role. Giacomo's exploitive plan would work because when an agent adopts a role in a group, that agent has no information about the kind of tasks it could be assigned. Tasks are rather created independently of roles, and are only subsequently associated with them.

In the example, the `plumber` agent reasonably will not have a plan to achieve the `do_a_very_strange_thing` goal. Consequently, when the corresponding obligation is created, it will not be fulfilled.

Given the above scenario, who could we consider accountable for the inevitable goal failure of `do_a_very_strange_thing`? Should the agent playing the `plumber` role be held accountable? The agent violated its obligation but could not have reasonably anticipated the goal's introduction, which effectively made achievement impossible. Should Giacomo be held accountable since he introduced an unachievable goal, however licit? Perhaps the system itself ought to bear the brunt of accountability since it permits such unfair behavior? The system, however, doesn't know agent capabilities and cannot consequently make a fair/unfair judgment call.

Listing 1.1 shows how the organization for the building-a-house scenario is defined in Moise. The file contains: the structural, functional, and normative specification.

Listing 1.1. Excerpt of the organization for building-a-house.

```
<organisational-specification id="house_contruction"
...
<structural-specification>
  <role-definitions>
    <role id="house_owner" />
    <role id="building_company" />
    <role id="plumber" >
        <extends role="building_company"/>
    </role>
    ...
  </role-definitions>
  <group-specification id="house_group">
    <roles>
      <role id="house_owner" min="1" max="1"/>
      <role id="plumber"     min="1" max="1"/>
      ...
    </roles>
    ...
  </group-specification>
</structural-specification>
<functional-specification>
  <scheme id="build_house_sch">
    <goal id="house_built">
      <plan operator="sequence">
        <goal id="site_prepared" ttf="20 minutes" />
        ...
        <goal id="plumbing_installed" ttf="20 minutes" />
        ...
      </plan>
    </goal>
    <mission id="management" min="1" max="1">
      <goal id="house_built"/>
    </mission>
    <mission id="prepare_site" min="1" max="1">
      <goal id="site_prepared" />
    </mission>
    <mission id="install_plumbing" min="1" max="1">
      <goal id="plumbing_installed" />
    </mission>
    ...
  </scheme>
</functional-specification>
<normative-specification>
  <norm id="n1" type="obligation" role="house_owner"
        mission="management" time-constraint="2 minutes" />
  ...
  <norm id="n8" type="obligation" role="plumber"
        mission="install_plumbing" />
  ...
</normative-specification>
</organisational-specification>
```

Line 7, for instance, defines a `plumber` role that is included in the `house_group` group at line 15. After the structural specification, we find the functional specification with a `build_house_sch` scheme. Line 37 defines an `install_plumbing` mission composed of the `plumbing_installed` goal. Finally, in the normative specification, norm `n8` binds the `plumber` role to the previously described mission. It's important to notice that this definition could change at runtime; in particular new schemes could be dynamically generated and, for instance, associated with the `house_group` that will become responsible for them.

Once the building phase is started, Giacomo creates a GroupBoard artifact, called `hsh_group`, following the XML specification of the `house_group`. GroupBoard artifacts are used to manage the lifecycle of specific group of agents. After that it adopts the role `house_owner`, and asks the auction winners (see [6] for explanations about the auction) to adopt the corresponding roles (`!contract_winners`). Finally, after all agents have adopted their roles and the group is ready, a SchemeBoard artifact called `bhsch` is created to manage the execution of the `build_house_sch` social scheme.

When the company agents receive the request sent by Giacomo, they adopt the roles by acting on the group artifact. From that moment on they could be asked (i.e. obliged) to commit to some missions according to the normative specification. This phase is needed in order to form the group which will become responsible of the scheme. For instance, agent `companyA` could be asked to commit to `install_plumbing` with an obligation of the form `obligation(companyA, n8, committed(companyA, install_plumbing, bhsch), ...)`. Norm `n8` is, indeed, the norm that binds the `plumber` role with `install_plumbing` in the normative specification. When the group is well-formed, agents inside it can be obliged to achieve the related goals. Indeed, the main purpose of the `SchemeBoard` artifact is to keep track of which goals are ready to be pursued and create obligations for the agents accordingly. For instance, let's assume the `plumbing_installed` goal is ready to be pursued; an obligation `obligation(companyA, ..., achieved(bhsch, plumbing_installed, companyA),...)` will be generated, provided that the `companyA` agent is playing the `plumber` role. Such obligations are observed by the agents and the corresponding goals are automatically created. Listing 1.2 shows an excerpt of the `companyA` agent. The obligation creates the goal which is then achieved following the plan of line 20. As soon as other goals are ready to be pursued, new obligations are created.

Listing 1.2. Excerpt of code of the `companyA` agent.

```
1  ...
2  task_roles("Plumbing", [plumber]).
3  +!contract(Task,GroupName)
4    : task_roles(Task,Roles) <-
5    ...
6    lookupArtifact(GroupName, GroupId);
7    for (.member(Role, Roles)) {
8      adoptRole(Role)[artifact_id(GroupId)];
9      focus(GroupId)
10   }.
11 +obligation(Ag,Norm,committed(Ag,Mission,Scheme),Deadline)
12   : .my_name(Ag) <-
13   commitMission(Mission)[artifact_name(Scheme)].
14 +obligation(Ag,Norm,achieved(Scheme,Goal,Ag),Deadline)
15   : .my_name(Ag) <-
16   ...
17   !Goal[scheme(Scheme)];
18   ...
19   goalAchieved(Goal)[artifact_name(Scheme)].
```

```
+!plumbing_installed   // the organisational goal
                          (created from an obligation)
  <- installPlumbing.  // simulates the action
```

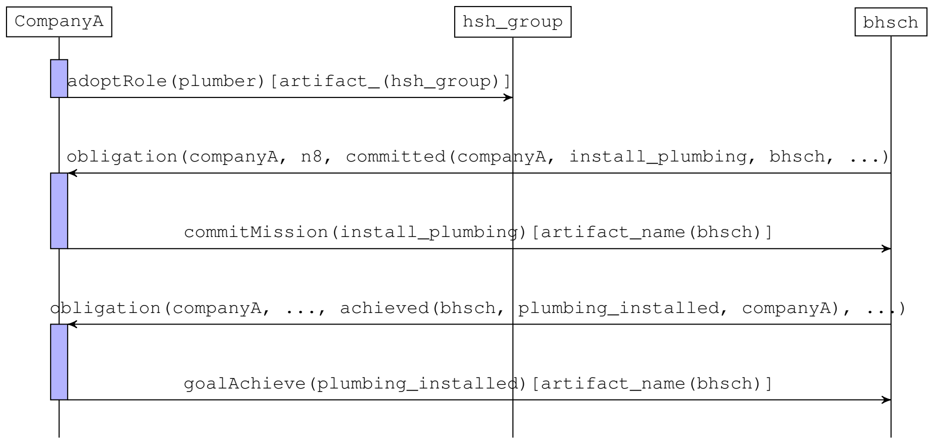
CompanyA
hsh_group
bhsch
adoptRole(plumber)[artifact_(hsh_group)]
obligation(companyA, n8, committed(companyA, install_plumbing, bhsch, ...)
commitMission(install_plumbing)[artifact_name(bhsch)]
obligation(companyA, ..., achieved(bhsch, plumbing_installed, companyA), ...)
goalAchieve(plumbing_installed)[artifact_name(bhsch)]

Fig. 1. Interaction between the `companyA` agent and the organization in the *building-a-house* example.

Figure 1 reports the general interaction pattern, concerning role adoption and mission distribution, instantiated on the `companyA` agent and plumbing. As underlined by the authors, *"[a] main advantage of this approach is that by simply changing the scheme specification (which can be done by the designer or by the agents themselves) at very high level, [...] we will change the overall behavior of the agent team without changing a single line of their code. [...] This artifact also manages the state of the obligations, checking, for instance, their fulfillment or violations. This feature is very useful for Giacomo who wants to monitor the execution of the scheme to ensure the house is built correctly and on time"*. Now, let's suppose Giacomo is dishonest and wants to achieve some tasks, that are not related to house construction, by assigning them to the player of plumber role. In particular, let's suppose he wants to delegate a `do_a_very_strange_thing` goal to the agent who is playing the `plumber` role (see Listing 1.3).

Listing 1.3. Organization specification involving `do_a_very_strange_thing`.

```
<functional-specification>
  <scheme id="build_house_sch">
    <goal id="house_built">
      <plan operator="sequence">
        ...
        <goal id="do_a_very_strange_thing" ... />
        ...
      </plan>
    </goal>
    ...
    <mission id="install_plumbing" min="1" max="1">
      <goal id="do_a_very_strange_thing" />
    </mission>
    ...
```

```
15   </scheme>
16 </functional-specification>
```

The only two lines that have been modified are Lines 6 and 12. This modification is licit even if the group that will be responsible for the execution has been already created. In fact, the GroupBoard artifact and the SchemeBoard artifacts are created in different moments. The problem here is that the role definition given in the structural specification of the organization says nothing about the kind of capabilities (or requirements) an agent should have in order to play the given role. Similarly, it is not specified what kind of tasks could be assigned to the agent.

3 A Characterization of Organizational Accountability

Our interest in accountability primarily lies with its application as a *design property* [3]; that is, we adopt the type of accountability that might be defined as *"an institutional relation or arrangement in which an agent can be held to account by another agent or institution"* [8]. Throughout our discussion, we will make use of the term, *forum*, which is an investigative body that evaluates and passes judgment on agents. As a design property, we consider integral the various steps to an accountability-as-a-mechanism relationship as described in [8]: a forum must receive all information, including all causal actions, regarding a given situation under scrutiny, the forum must be able to contextualize actions to understand their adequacy and legitimacy, and finally the forum must be able to pass judgment on agents. Our goal lies in automating the entire process, that is, to create a structure that creates and collects contextualized, integral information so that accountability can be determined from any future institutional state.

One of the key difficulties in realizing our goal lies with the tricky notion of *contextualized action*. In our own societies, contextualizing might entail an examination of circumstances: for example, what should have a person done, why didn't she/he do that, what impact did her/his actions have, and given what the person had to work with, did she/he act in an exemplary fashion? The same process in a MAS would be guided by the same type of questions, though in order to facilitate their answers, we need to make use of different structures. In particular, we need structures that allow assessing who is accountable without actually infringing on the individual and private nature of agents. We can determine action impact or *significance* by identifying the amount of disruption it causes in terms of other agents and/or work affected.

We identify the following necessary-but-not-sufficient principles a MAS must exhibit in order to support accountability determinations.

Principle 1 *All collaborations and communications subject to considerations of accountability among the agents occur within a single scope that we call* organization.

In a word, situatedness. Accountability must operate in a specific context because individual actions take on their significance only in the presence of the larger whole. What constitutes a highly objectionable action in one context could instead be worthy of praise in another. Correspondingly, a forum can only operate in context and an agent's actions must always be contextualized. The same role in different contexts can have radically diverse impacts on the organization and consequently on accountability attribution. When determining attribution, thus, an organization will only take into account interactions that took place inside its boundaries.

Placing an organizational based limit on accountability determinations serves multiple purposes. It isolates events and actors into more manageable pieces so that when searching for causes/effects, one need not consider all actions from the beginning of time nor actions from other organizations. Agents are reassured that only for actions within an organization will they potentially be held accountable. Actions, thanks to agent roles, also always happen in context.

As illustrated in [10], accountability attribution consists in a rather complex process involving an investigative forum to assess the situation and evaluate who is accountable for what and to what degree[3]. The indispensability of the forum becomes clear in our societies in which intrigue and complex motivations come to bear. Luckily, a MAS greatly simplifies the matter, and our task takes the form of ensuring all possible actions are accounted for and categorized with respect to accountability, whose attribution will occur post-execution. The influence of an unrealized goal on its resulting mission failure determine the degree of accountability. For instance, should two agents fail to bring about their goals, leading to the failure of another, both agents would bear half the accountability for the consequent failure. Should only one agent cause the failure of a goal, that agent would bear the full brunt of the accountability.

To adequately account for accountability by categorizing action, we must deal with two properties within a given organization: 1) an agent properly completes its tasks and 2) an agent does not interfere with the tasks of others. The principles below deal more explicitly with the first property; that is, how to ensure that agents complete their tasks in a manner fair for both the agents and the organization. The second property is also partially satisfied in our discussion by ensuring that, in the presence of goal dependencies, the first agent in sequence not to complete its goal will bear accountability, not only for its incomplete goal, but for all dependent goals that will consequently remain incomplete. That is, should an agent be responsible for a goal on whose completion other agents wait, and should that agent not complete its goal, then it will be accountable for its incomplete goal and for that goal's dependents as well.

Principle 2 *An agent can enroll in an organization only by playing a* role *that is defined inside the organization.*

As an organizational and contextual aid to accountability, roles attribute social significance to an agent's actions and can, therefore, provide a guide to the severity of non-adherence.

Principle 3 *An agent willing to play a role in an organization must be aware of all the powers associated with such a role before adopting it.*

Following the tradition initiated by Hohfeld [17], a power is "one's affirmative 'control' over a given legal relation as against another." The relationship between powers and roles has long been studied in fields like social theory, artificial intelligence, and law.

[3] In the present proposal accountability is crisp and either holds or does not hold.

Here we invoke a knowledge condition for an organization's agents, and stipulate that an agent can only be accountable for exercising the powers that are publicly given to it by the roles it plays. Such powers are, indeed, the means through which agents affect their organizational setting. An agent cannot be held accountable for unknown effects of its actions but, rather, only for consequences related to an agent's known place in sequences of goals. On the other hand, an agent cannot be held accountable for an unknown goal that the organization attaches to its role, and this leads us to the next principle.

Principle 4 *An agent is only accountable, towards the organization or another agent, for those goals it has explicitly accepted to bring about.*

An organization may not obligate agents to complete goals without prior agreement otherwise we find ourselves in the unfortunate previously discussed scenario in which an organization can insert goals irrelevant to a given role like `do_a_very_strange_thing` in an agent's obligations. In other words, an organization must always communicate to each agent the goals it would like the agent to pursue.

Principle 5 *An agent must have the leeway for putting before the organization the provisions it needs for achieving the goal to which it is committing. The organization has the capability of reasoning about the requested provisions and can accept or reject them.*

Notice that with this principle we diverge from considerations in the field of ethics regarding accountability in the presence of causal determinism [15,9], where even in the absence of alternate possibilities humans can be morally responsible thanks to the significance of the choice to act. Finding the conversation fundamentally shifts when speaking of software agents, we consequently conclude that accountability is not attributable in the presence of impossibilities. Correspondingly, agents must be able to stipulate the *conditions* under which a given goal's achievement becomes possible, i.e. the agent's requested *provisions*. The burden of discovery for impossibilities, therefore, rests upon an agent collective who announce them by their combined silence for a given goal. That is, a goal becomes effectively impossible for a group of agents should no agent stipulate a method of achievement. Conversely, an agent also declares a goal possible the moment it provides provisions to that goal. Should an uniformed agent stipulate insufficient provisions for an impossible goal that is then accepted by an organization, that agent will be held accountable because by voicing its provisions, it declared an impossible goal possible. The opportunity to specify provisions, therefore, is fundamental in differentiating between impossibilities and possibilities.

To illustrate the need for provisions to model accountability, we can imagine an organization consisting of two members: one to prepare a wall, *wall-preparer*, and another who paints the wall, *painter*. Their organization would give both access rights to the wall and attribute to *wall-preparer* the goal of prepping the wall, and to *painter* the goal of painting the wall. Without the possibility of stipulating when goals are possible, perhaps *wall-preparer* fulfills its goal but whimsically paints a black stripe down the middle. Unfortunately, now *painter* has inadequate materials and cannot realize its goal. *Wall-preparer* made *painter's* goal impossible. The impossibility, however, only came about at runtime. Should *painter* stipulate provisions for its goal, it effectively qualifies possibility, permitting accountability to work by guaranteeing an absence of impossibilities.

4 The ADOPT Accountability Protocol

We turn now to a MAS design-phase application of the above-mentioned accountability principles. Chopra and Singh explored a similar approach of design-phase accountability in [12]. In their work, Chopra and Singh suggest that an actor can legitimately depend on another to make a condition become true only when such a dependency is formalized in an *institutionalized expectation*, whose structure describes expectations one actor has of another and whose inherently public nature exerts normative power. To tackle accountability as a design property, Chopra and Singh introduce the notion of *accountability requirement* as a special case of institutionalized expectation. An accountability requirement is a relation involving two principals, an account giver (a-giver) and an account taker (a-taker). The a-giver is accountable to the a-taker regarding some conditional expectation; namely, the expectation involves an antecedent condition and a consequent condition. Usually, the consequent condition is pursued only when the antecedent condition is true. In principle, if an accountability requirement is violated, the a-taker has a legitimate reason for complaint. The notion of accountability requirement can be further refined in terms of *commitments*, *authorizations*, *prohibitions*, and *empowerments* [12]. Each of these relations has specific implications in terms of who is accountable and for what reason. It is worth noting that an a-giver is normally accountable for a specific condition towards the whole group of agents in a MAS. That is, in an agent society, agents are accountable for their actions towards the society as a whole. Rather than creating an accountability requirement between each possible pairs of a-giver and a-taker, it is convenient to adopt the perspective by Chopra and Singh; namely, considering both the agents and the organization as principals, between which mutual expectations can be defined.

In other words, an organization is considered as a *persona iuris* [12], a legal person that can be the a-giver or a-taker of an accountability requirement, as any other principal represented by an agent. In addition, an organization will also be the conceptual means through which complex goals are articulated in terms of subgoals and distributed among a set of *roles*. An organization is, therefore, a design element that allows one to specify: (1) what should be achieved by the MAS (i.e., the organizational goals) and (2) what roles are included in the organization and with what (sub)goals. As far as accountability is concerned, an organization that shows the above features naturally satisfies Principles 1 – 3.

Our intuition is that in order to obtain accountability as a design property of a MAS, the agents who are willing to be members of an organization enroll in the organization by following a precise *accountability protocol*. The organization provides the context in which accountability requirements are defined. To define such an accountability protocol, we rely on the broad literature about commitment-based protocols and focus our attention on the accountability requirements that can be expressed as (practical) commitments. Commitments have been studied at least since the seminal works by Castelfranchi [11] and Singh [23]. A social commitment is formally represented as $\mathsf{C}(x, y, p, q)$, where x is the debtor (a-giver, in our case), that commits to the creditor y (a-taker) to bring about the consequent condition q should the antecedent condition p hold. From the accountability point of view, the a-giver is accountable when the antecedent becomes true, but the consequent is false.

The gist of the accountability protocol is to make explicit the legal relationships between the agent and the organization. These are expressed as a set of (abstract) commitments, directed from organizational roles towards the organization itself, and vice versa. The first step captures the adoption of a role by an agent. Let $pwr_{i,1}, \ldots, pwr_{i,m}$ be the powers that agent Ag_i, willing to play role R_i, will get. Ag_i will commit towards the organization to exercise the powers, given to it by the role, when this will be requested by the legal relationships it will create towards other agents. In this way, the agent stipulates awareness of the powers it is endowed with, becoming accountable, not only towards some other agent in the same organization but also towards the organization itself, of its behavior:

$$\begin{aligned} cpwr_{i,1} &:: \mathsf{C}(Ag_i, Org, \mathsf{C}(Ag_i, Z_1, pwr_{i,1}), pwr_{i,1}) \\ &\ldots \\ cpwr_{i,m} &:: \mathsf{C}(Ag_i, Org, \mathsf{C}(Ag_i, Z_m, pwr_{i,m}), pwr_{i,m}) \end{aligned}$$

above Z_j, $j = 1, \ldots, m$ represent some roles or some (not necessarily different) agents in the organization. These commitments represent the fact that, from an accountability-based point of view, an agent, when exercising a power because of a social relationship with some other agents, has some duties towards the social institution which provides that power, too. Indeed, when an employee is empowered by a manager to perform a given task on behalf of the company, the result is not only a commitment of the employee with the manager, but also a commitment of the employee with the company. An agent willing to play a role is expected to create a commitment that takes the form:

$$cpwr_{R_i} :: \mathsf{C}(Ag_i, Org, \mathsf{accept\_player}_{Org}(Ag_i, R_i), cpwr_{i,1} \wedge \cdots \wedge cpwr_{i,m})$$

where $\mathsf{accept\_player}_{Org}(Ag_i, R_i)$ is a power of the organization to accept agent, Ag_i, as a player of role R_i.

Org, then, has the power to assign goals to the agents playing the various roles through $assign_{Org}$. This is done through the creation of commitments by which the organization promises to assign some goal to some agent should the agent accept to commit to pursue the goal:

$$\begin{aligned} cass_{i,1} &:: \mathsf{C}(Org, Ag_i, cg_{i,1}, \mathsf{prov}_{i,1} \wedge \mathsf{assign}_{Org}(Ag_i, goal_{i,1})) \\ &\ldots \\ cass_{i,n} &:: \mathsf{C}(Org, Ag_i, cg_{i,n}, \mathsf{prov}_{i,n} \wedge \mathsf{assign}_{Org}(Ag_i, goal_{i,n})) \end{aligned}$$

Above, $cg_{i,k=1,\ldots,n}$ denote the commitments by whose creation the agent explicitly accepts the goals and possibly asks for provisions $\mathsf{prov}_{i,k=1,\ldots,n}$. Here, $goal_{i,k}$ is a goal the organization would like to assign to the agent Ag_i. The antecedent condition of $cg_{i,k}$ has the shape $prov_{i,k} \wedge assign_{Org}(Ag_i, goal_{i,k})$, where $prov_{i,k}$ stands, as said, for a provision the agent requires for accomplishing the task, and the consequent condition has the shape $achieve_{Ag_i}(goal_{i,k})$:

$$\begin{aligned} cg_{i,1} &:: \mathsf{C}(Ag_i, Org, \mathsf{prov}_{i,1} \wedge \mathsf{assign}_{Org}(Ag_i, goal_{i,1}), \mathsf{achieve}_{Ag_i}(goal_{i,1})) \\ &\dots \\ cg_{i,n} &:: \mathsf{C}(Ag_i, Org, \mathsf{prov}_{i,n} \wedge \mathsf{assign}_{Org}(Ag_i, goal_{i,n}), \mathsf{achieve}_{Ag_i}(goal_{i,n})) \end{aligned}$$

Provisions are to be instantiated with those prerequisites that Ag_i discloses as necessary for it to complete its job and that Org is expected to provide. On the agent side, these commitments are the means through which the agent arranges the boundaries of its accountability within the organization. For instance, *painter*, in our example above, is an agent hired in a painting organization including also *wall-preparer*. A provision for *painter* to paint a wall could be *wall-prepared*, a condition that is to be achieved by another agent from the same organization, and that appears in the accountability requirements of its role. Should *wall-preparer* behave maliciously (as in our example), *painter* would not be accountable for not painting the wall as provision *wall-prepared* would be missing. On the organization side, provisions are part of the information used to decide whether to assign the goal to the agent (the internal decision processes of an organization are outside the scope of the paper). An agent becomes obliged to achieve a goal only after this assignment so as to not violate the accountability requirement. Finally, $achieve_{Ag_i}(g_{i,j})$ denotes that goal $goal_{i,j}$ is achieved.

After these premises, we can now introduce the protocol that regulates the enrollment of an agent, Ag_i, in an organization, Org, as a player of role, R_i, and the subsequent assignment of goals to Ag_i carried out by Org.

(1) $\mathsf{create}(cpwr_{R_i})$
(2) $\mathsf{accept\_player}_{Org}(Ag_i, R_i)$
(3) $\mathsf{create}(cpwr_{i,1}), \dots, \mathsf{create}(cpwr_{i,m})$
(4) $\mathsf{create}(cass_{i,k}), k = 1, \dots, n$
(5) $\mathsf{create}(cg_{i,k}), k = 1, \dots, n$
(6) $\mathsf{assign}_{Org}(Ag_i, goal_{i,k}), k = 1, \dots, n$
(7) $\mathsf{prov}_{i,k}, k = 1, \dots, n$
(8) $\mathsf{achieve}_{Ag_i}(goal_{i,k}), k = 1, \dots, n$

An agent Ag_i, willing to play role R_i, makes the first step by creating the commitment, $cpwr_{R_i}$ (1). By doing so it proposes itself as role player. It is worth noting that the creation of $cpwr_{R_i}$ is possible only as a consequence of Principle 3, by which an organization must disclose the powers associated with its roles. The organization is free to decide whether to accept an agent as role player (2). In case of acceptance the agent creates the commitments by which it becomes accountable with the organization of the use of its powers (3). Step (4) allows the organization to communicate the goals it wishes to assign to the agents. The agents are expected to accept them by creating the corresponding commitments of Step (5), thereby knowing which goals it may be asked to achieve at Step (6). Steps (7) and (8) respectively allow the organization to satisfy the provisions, and the agent to communicate goal achievement.

Principle 1 finds an actualization in the fact that all the mentioned commitments are created within a precise organization instance. When Org accepts Ag_i as a player for role R_i, the enrollment of the agent is successfully completed. After this step, the agent operates in the organization as one of its members. This satisfies Principle 2, for which an agent is a member of an organization only when it plays an organizational role. Principles 4 and 5 find their actualization in terms of the commitments $cg_{i,k}$'s. Principle 4 demands that an agent is accountable only for those goals it has explicitly accepted to bring about. The creation of one of the commitments $cg_{i,k}$ represents the acceptance of being responsible, and hence accountable, for the goal occurring in the commitment consequent condition. Principle 5 states that an agent must have the leeway to negotiate its own duties, which we obtain in two ways. First, the agent creates its own commitments, which means that the mission commitments might cover just a subset of the goals. Second, the agent can make explicit provisions for each role goal.

4.1 Verifying ADOPT

Driven by the five fundamental principles we have identified, we have proposed an accountability protocol to achieve accountability as a design property in a MAS. We now wish to verify that the proposed protocol actually adheres to the five principles. Notably, in this paper we have used commitments as a means for specifying *accountability requirements*; that is, for specifying what a principal (either an agent or the whole organization) can legitimately expect from others, and vice versa. This choice has some important design consequences. In order to create the commitment $cpwr_{R_i}$, an agent willing to play role R_i must be aware both of the organization Org and of the role itself within Org together with the powers $pwr_{i,1}, \ldots, pwr_{i,n}$ associated with R_i. This means that: 1) the organization must exist, 2) roles must be defined in the context of an organization, and 3) powers associated with roles must be known at the time of role enactment. When these elements are all known to an agent before joining an organization, the system implicitly satisfies the accountability principles 1, 2, 3, and 5. In other words, these principles are structurally satisfied by the adoption of commitments as a means to represent accountability requirements. The only principle that is still to be verified is Principle 4: an agent is only accountable for those goals for which it has taken an explicit commitment. The verification of this principle demands consideration of the dynamics of the accountability protocol in order to check whether such a principle is ever violated. We do this by translating Principle 4 into a set of CTL formulae and by verifying with a model checker whether the protocol satisfies these properties.

For the sake of discussion, we present here the CTL formulae in an abstract way, assuming the existence of only one agent willing to play the unique role, which can exert only one power, in a given organization. Provisions are not addressed explicitly, but the following discussion can be extended to treat them as well. Let us assume that the agent has already created the commitment, $cpwr_{R_i}$. As noted above, this is only the first step of enactment. In fact, to complete the enactment phase, the organization has to accept the agent. Only after this second step is the agent obliged to commit to the powers associated with the role. The following two properties capture this aspect of Principle 4.

$$\mathsf{AG}(enactment \rightarrow \mathsf{AF}(commit\_pwr)) \tag{1}$$

$$\mathsf{A}(\neg commit\_pwr \;\mathsf{U}\; enactment) \tag{2}$$

$$\mathsf{AG}(enactment \rightarrow \mathsf{AF}(\mathsf{A}(commit\_pwr \;\mathsf{U}\; exert\_goal))) \tag{3}$$

Formula (1) specifies that whenever the enactment occurs ($enactment$) the agent will create the commitment to exert the power pwr as and when it will be expected by another principal Z (i.e., either another agent in the organization, or the organization itself). Formula (2), on the other hand, specifies that the agent will not commit to a exert a power until the organization completes enactment through acceptance. The second formula is required because we want to avoid situations in which an agent commits to use a power until it is endowed with the power itself. Finally, Formula (3), means that an agent remains committed to use the powers until it will actually need to use them.

So far, we have just modeled the properties that a proper role enactment phase must satisfy. The key aspect of Principle 4, however, is about the actual achievement of an organizational goal. Goals are issued by Org dynamically according policies which fall outside the scope of our discussion. What we want to verify is that an agent commits to goals that, although assigned by Org, have been previously accepted by the agent and are achievable with the powers that Org endows the agent with. This is expressed by the following CTL formulae.

$$\mathsf{A}(\neg commit\_goal\ \mathsf{U}\ publish\_goal) \tag{4}$$
$$\mathsf{AG}(commit\_goal\ \rightarrow \mathsf{AF}\ assign\_goal) \tag{5}$$
$$\mathsf{AG}(assign\_goal \rightarrow \mathsf{AF}(exert\_pwr)) \tag{6}$$

Formula (4) means that an agent will not commit to a goal until it is published by the organization. Note, however, that when the organization publishes a goal, the agent has the free choice of accepting the goal. However, when the agent commits to a goal previously published by the organization, the organization is then obliged to assign the goal to the agent, this is modeled in Formula (5). Finally, formula (6) means that whenever a goal is assigned to an agent, the agent will attempt to achieve the goal by exerting the power it has previously committed to. Of course, in practical situations, the agent may fail to achieve the goal, but from the point of view of ADOPT, and of Principle 4, the agent has done its job if it has, at least, tried to achieve the goal by using its powers. Determining the causes of a failed goal could involve various forms diagnostic reasoning, such as [20,21], which, although relevant for the accountability point of view, are left to future work. It is possible to show that our accountability protocol satisfies these CTL properties.

4.2 Applying ADOPT to the Building-a-House Scenario

We now show how the ADOPT protocol can be applied to the *building-a-house* example introduced above. First of all, ADOPT requires the existence of an organization where roles and powers associated with roles are disclosed. Here we focus on the role `plumber` and on its power *install_plumbing*. Then, an agent, here `companyA`, willing to play the role at issue should create the following two commitments, as a first step of the enactment phase:

$cpwr_p ::\ \mathsf{C}(companyA, Org, \mathsf{accept\_player}_{\texttt{Org}}(companyA, plumber), cpwr_{inst\_p})$
$cpwr_{inst\_p} ::\ \mathsf{C}(companyA, Org, c_p, install\_plumbing)$

By creating the nested commitment on top, $cpwr_p$, the company accepts to be accountable with the organization for the power to install the plumbing within the organization itself, if it is accepted as `plumber`. Acceptance is completed by the creation of a set of specific commitments, each one concerning a single power. Here we have only one such commitment ($cpwr_{inst\_p}$) because the role gives the agent only one power. The antecedent condition c_p of such a commitment amounts to $\mathsf{C}(companyA, Owner, install\_plumbing)$ that binds the company towards a role $Owner$ (of the future house) in the same organization. Thus, the nested commitment raises accountability for installing the plumbing to the organizational level.

Then, the organization dynamically assigns goals to its members; i.e., agents who were accepted as role players. In particular, ADOPT assumes that an organization will assign a goal to an agent only if the agent is aware of the goal and accepted it. Also in this case, we use commitments to formalize the relationship between the organization and the agent. Specifically, in our example `Org` assigns the goal to perform *install_plumbing* by means of the commitment $cass_{ip}$:

$$\begin{aligned} cass_{ip} &:: \mathsf{C}(Org, companyA, cg_{g\_ip},\\ &\qquad \mathsf{prov}_{g\_ip} \wedge \mathsf{assign}_{Org}(companyA, install\_plumbing))\\ cg_{g\_ip} &:: \mathsf{C}(companyA, Org, \mathsf{prov}_{g\_ip} \wedge \mathsf{assign}_{Org}(companyA, install\_plumbing),\\ &\qquad \mathsf{achieve}_{companyA}(install\_plumbing)) \end{aligned}$$

In words, `Org` commits to assign the goal to `companyA` and to supply the related provisions (prov_{g_ip}) if the agent takes a commitment (cg_{g_ip}) to pursue the goal, given those provisions and if the goal is assigned to it. Here provisions are assumed to be the result of a negotiation phase which is outside the scope of the paper; of course, it may be possible that no provision is requested. If `companyA` accepts the mission (i.e., goal and provisions), it creates commitment cg_{g_ip}, and this will detach commitment $cass_{ip}$. This is the pivotal aspect of the ADOPT proposal: `companyA` becomes accountable towards `Org` because it assumes voluntarily the responsibility of bringing about a goal in case that goal will ever be assigned to it. `Org` is now expected to assign `companyA` goal *install_plumbing* for discharging its commitment. In the example, `Org` will also have to bring about provisions in order to discharge its commitment towards `companyA`.

Now, `companyA` is obliged by the detached commitment cg_{g_ip} to bring about the goal *install_plumbing*. This goal can be achieved by using the power `companyA` has acquired, and has committed to use, when it joined the organization by issuing the commitments $cpwr_p$ and $cpwr_{inst\_p}$ during the enactment phase. If the goal is not achieved because power *install_plumbing* is not exerted, `companyA` can be held accountable for the violation of $cpwr_{inst\_p}$ and cg_{g_ip}. Namely, `companyA` can be held accountable because it didn't even attempt to bring about a task it had committed to when it had adopted the role. On the other hand, if the organization assigns an unexpected goal to `companyA`, let us say goal, *do_a_very_strange_thing*, this does not detach any commitment and, consequently, `companyA` is not obliged to do anything. It is worth noting that thanks to the ADOPT protocol we can immediately identify the behavior of the organization as a violation of the accountability property. In fact, because of Principle 4, the organization is not authorized to issue goals to agents that are not committed to bring about those goals.

5 Discussion

Our work with JaCaMo highlights a conceptual challenge in the concept of role and a role's central place in responsibility and accountability (in the form of "role-following responsibility") as illustrated by [13]. To a certain degree, decoupling a role from an organizational execution essentially negates the role's function to limit its operational domain. As illustrated with building-a-house, without prior agreement of what exactly a role means in a particular organizational context, we can force a role to mean whatever we want so long as the language matches. The consequent dynamism of roles makes automatic considerations of accountability impossible to conclude. In our construction of computational accountability, roles represent a division of responsibility and pattern of interaction that serve the investigative forum to assign accountability.

The accountability protocol allows design-phase incorporation of accountability (1) by excluding that the organization changes the goals assigned to roles after agents enacted them and (2) by allowing agents to make their provisions explicit. As one way to enforce a behavior that respects the protocol, one could modify JaCaMo's conceptual model and implementation so that it follows the five principles. The modification would be possible since JaCaMo relies on obligations, which can be used to represent detached commitments. Another way is to introduce proper monitors that, if needed, can check protocol adherence. This calls for the realization of a kind of artifact that can monitor the interaction, represent the social state (made of the existing commitments), and track its evolution. This kind of system could be realized by means of 2COMM [1,2].

The resulting accountability-supporting organization has affinities with the model of social structures defined in [14], which describe a *whole* made up of *parts* that are organized by specific relationships. Wholes and parts are all entities, and parts can be wholes themselves. The social structure has *causal properties* (that is, it can affect the world) in its own right. Such properties (also known as powers), synchronically emerge only with the constitution of a social structure. Even in the presence of all individual members, if the structure's characterizing relationships are absent, so too are the previously mentioned properties. In this framework, it is easy to see that accountability is, indeed, an emergent property of a social structure (organization). Like all emergent causal properties of a social structure, it co-exists with the causal powers of its parts (the agents), whose acts are affected by the ways in which they are organized, so generally events are multiply determined.

If we adapt the approach to roles developed in [4,5] in which roles essentially define an organization, accountability takes on functional implications for the very definitional existence of the organization. Should some roles remain unfulfilled, an organization would correspondingly find itself in definitional crisis. As illustrated in [16], role fulfillment means continual realization of role relationships, that is, a role's duties and obligations. Accountability allows an organization some recourse in crisis and a method of expressing the relative importance its roles play. Armed with the knowledge of relative responsibility and therefore importance in the collective, an organization enables role-playing agents to make informed decisions should conflicts arise and to make their own cost/benefit analysis should an agent not wish to not perform its function.

A mechanism based on commitments presents numerous conceptual advantages for accountability. An agent is able to specify the exact social context in which it can fulfill the specified goal, g. It effectively announces to the organization, Org, that should its requirements become true, it will be accountable for fulfilling g. Essentially the commitments require pre-execution knowledge of expectations and requirements both on the part of the organization and of the agent, which satisfies accountability's foreknowledge requirement. Commitments can therefore provide indications of responsibility, as a pre-execution assignment, which will then, thanks to the exhaustive definitions of pre and post conditions, provide a direct mapping to accountability post execution. Since the agent, Ag, by design creates the commitment to the organization, the agent, not the organization, specifies its requirements to satisfy the goal, g. Casual determinism consequently cannot manifest because agent Ag stipulates the exact social circumstances in which it can operate and realize g. Moreover, role relationships become explicit through the provision stipulation, which will later provide a basis for role-adherence determination. The commitment structure therefore provides the necessary characteristics for beginning to speak of accountability.

Acknowledgments. This work was partially supported by the *Accountable Trustworthy Organizations and Systems (AThOS)* project, funded by Università degli Studi di Torino and Compagnia di San Paolo (CSP 2014). The authors warmly thank the reviewers for their constructive and helpful comments which helped revising the paper.

References

1. Matteo Baldoni, Cristina Baroglio, and Federico Capuzzimati. A Commitment-Based Infrastructure for Programming Socio-Technical Systems. *ACM Transactions on Internet Technology*, 14(4):23:1–23:23, December 2014.
2. Matteo Baldoni, Cristina Baroglio, Federico Capuzzimati, and Roberto Micalizio. Commitment-based Agent Interaction in JaCaMo+. *Fundamenta Informaticae*, 2017. To appear. Available at `http://www.di.unito.it/~argo/papers/2017_FundamentaInformaticae.pdf`.
3. Matteo Baldoni, Cristina Baroglio, Katherine M. May, Roberto Micalizio, and Stefano Tedeschi. Computational accountability. In *Proceedings of the AI*IA Workshop on Deep Understanding and Reasoning: A Challenge for Next-generation Intelligent Agents 2016*, volume 1802 of *CEUR Workshop Proceedings*, pages 56–62. CEUR-WS.org, 2017.
4. Matteo Baldoni, Guido Boella, and Leendert W. N. van der Torre. Interaction between objects in powerJava. *Journal of Object Technology*, 6(2):5–30, 2007.
5. Guido Boella and Leendert van der Torre. The ontological properties of social roles in multi-agent systems: definitional dependence, powers and roles playing roles. *Artificial Intelligence and Law*, 15, 2007.
6. Olivier Boissier, Rafael H. Bordini, Jomi F. Hübner, Alessandro Ricci, and Andrea Santi. Multi-agent oriented programming with JaCaMo. *Sci. Comput. Program.*, 78(6):747–761, 2013.
7. Rafael H. Bordini, Jomi F. Hübner, and Michael Wooldridge. *Programming multi-agent systems in AgentSpeak using Jason*, volume 8. John Wiley & Sons, 2007.
8. Mark Bovens, Robert E. Goodin, and Thomas Schillemans, editors. *The Oxford Handbook of Public Accountability*. Oxford University Press, 2014.
9. Matthew Braham and Martin van Hees. An anatomy of moral responsibility. *Mind*, 121(483), 2012.

10. Brigitte Burgemeestre and Joris Hulstijn. *Handbook of Ethics, Values, and Technological Design: Sources, theory, values and application domains*, chapter Designing for Accountability and Transparency: A value-based argumentation approach. Springer, 2015.
11. Cristiano Castelfranchi. Commitments: From individual intentions to groups and organizations. In *ICMAS*, pages 41–48. The MIT Press, 1995.
12. Amit K. Chopra and Munindar P. Singh. The thing itself speaks: Accountability as a foundation for requirements in sociotechnical systems. In *IEEE 7th Int. Workshop RELAW*, page 22. IEEE Computer Society, 2014.
13. Rosaria Conte and Mario Paolucci. Responsibility for societies of agents. *Journal of Artificial Societies and Social Simulation*, 7(4), 2004.
14. Dave Elder-Vass. *The causal power of social structures: emergence, structure and agency*. Cambridge Univ Press, 2010.
15. Harry G. Frankfurt. Alternate possibilities and moral responsibility. *The Jounral of Philosophy*, 66(23), 1969.
16. Nicola Guarino and Christopher Welty. Evaluating ontological decisions with OntoClean. *Communications of the ACM*, 45(2):61–65, 2002.
17. Wesley Newcomb Hohfeld. Some fundamental legal conceptions as applied in judicial reasoning. *The Yale Law Journal*, 23(1):16–59, 1913.
18. Jomi F. Hübner, Olivier Boissier, Rosine Kitio, and Alessandro Ricci. Instrumenting multiagent organisations with organisational artifacts and agents. *Autonomous Agents and Multi-Agent Systems*, 20(3):369–400, 2010.
19. Jomi F. Hubner, Jaime S. Sichman, and Olivier Boissier. Developing organised multiagent systems using the MOISE+ model: Programming issues at the system and agent levels. *Int. J. Agent-Oriented Softw. Eng.*, 1(3/4):370–395, 2007.
20. Roberto Micalizio and Pietro Torasso. Agent cooperation for monitoring and diagnosing a MAP. In *Multiagent System Technologies, 7th German Conference, MATES 2009, Hamburg, Germany, September 9-11, 2009. Proceedings*, pages 66–78, 2009.
21. Roberto Micalizio and Pietro Torasso. Cooperative monitoring to diagnose multiagent plans. *Journal of Artificial Intelligence Research*, 51:1–70, 2014.
22. Alessandro Ricci, Michele Piunti, Mirko Viroli, and Andrea Omicini. *Environment Programming in CArtAgO*, pages 259–288. Springer US, Boston, MA, 2009.
23. Munindar P. Singh. An ontology for commitments in multiagent systems:. *Artificial Intelligence and Law*, 7(1):97–113, 1999.