Abstract. Groups of collaborative agents within organizations need to achieve goals creating group awareness. This is necessary for groups to act as single entities. The notion of collective belief has been used extensively in formal models for collaborative activity to deal with group awareness. However, collective belief alone is neither sufficient nor effortless for group members to act collaboratively. In human organizations, the members of a group accept that certain states hold based on shared practices and beliefs of individual agents. These acceptances are formed even if some members of the group do not believe that the corresponding states hold. This paper distinguishes between individual beliefs and group acceptances in multi-agent systems in well-organized settings. It introduces state recognition recipes that drive groups within organizations to create common awareness, and thoroughly explains the exploitation of these recipes.

1 Introduction

The ability to interact with others is one of the fundamental characteristics of agency. However, to form collaborative groups and act as group members, the exchange of bit streams is not enough. Agents must share knowledge and practices and must be able to build and maintain models of their peers.

This paper deals with groups of agents within organized settings: groups follow a pre-specified organizational model stated in terms of roles. The objective of this research is to build multi-agent systems that form the digital analogue of human organizations and help humans to fulfill their responsibilities individually or in collaboration with other colleagues. Agents represent humans within the digital organization. Representatives play the roles that their human partners play within the human organization and have the same responsibilities.

Subsequently, referring to individual agents or groups of agents playing specific roles within organizations we mean representatives or groups of representatives that play the corresponding roles within digital organizations.

Participating in a group, agents must reconcile between their individual beliefs and group’s views, creating group awareness independently of their perceptual and cognitive abilities, permissions to access information sources, knowledge that they posses, preferences etc.
Theoretical models of collaborative decision making [2, 8, 12, 5] adopt the notion of collective or mutual belief to cope with group awareness. According to this notion, a group of agents collectively believe a state $\alpha$ if every individual in the group believes $\alpha$ and everyone in the group believes that everyone believes $\alpha$, and so on. This definition is well-grounded; however, in settings where information is inherently distributed and access restrictions to information sources apply, the views of a group cannot be based on the individual beliefs of all group members.

Furthermore, it is rather difficult to build systems based on the notion of collective belief, unless certain assumptions are made [4, Chapter 11]. Such assumptions are often implicit, affecting system’s openness and dynamic re-structuring: newcomers in a group must be informed about these assumptions, and the group may need to adjust its policies for group awareness when it is re-structured.

The above issues introduce the problem of representing and exploiting the policies for building and maintaining common awareness within groups. For instance, in certain settings, group members must be able to exploit a policy that states that the group shall accept something only if the majority of the group members believe it, although there may be some group members with a different opinion. More than policies, the above example reveals the necessity for agents to clearly distinguish between their beliefs and their acceptances as group members.

Concluding the above, based solely on the notion of collective belief, agents can not tolerate different beliefs towards group common awareness, leaving outside many interesting group settings.

This paper distinguishes between acceptances and beliefs [13, 11], and proposes state recognition recipes for the representation of group policies towards forming group acceptances.

As it is stated in [13], “one can accept something even if they feel it is false … (and) whereas acceptance is voluntary, and thus decidable at will, belief is not [1]”. Based on this distinction, we assume that group members form group-accepted beliefs or acceptances; these are accepted by the group as a whole, even if there are individuals that do not believe them.

Section 2 of this paper presents a simple model for specifying organizational structures. Section 3 introduces the notion of responsibility and section 4 presents the state recognition recipes (r-recipes), i.e. the representations of group policies for forming acceptances. Section 5 presents how r-recipes are exploited towards forming acceptances and finally, section 6 concludes the paper.

2 Organizational structure

This section presents a model for specifying the structure of an organization. A complete model that captures every aspect of an organization is out of the scope of this paper.

A central construct in the model is that of the “role”. A role serves as a prototype that specifies the normative behaviour of an individual or of a set of individuals that form a group. Examples of roles are “company” and “department”. Generally, roles can have several constituents and can be related to each other with various relations.
In this paper we assume that each role comprises responsibilities and recipes, and that roles are interrelated via the relation “contains”. Using the “contains” relation between roles, a role aggregation hierarchy is created. Roles are distinguished into atomic roles, which do not contain other roles, and composite roles, that contain at least one role. The roles layer in Fig. 1 shows a small example of a role aggregation hierarchy. Roles customer, “seller”, “engineer” and “manufacturer” are atomic roles, while roles “company”, “customer department”, “engineering department” and “manufacturing department” are composite roles. According to this specification, a company has three types of departments: customer departments, engineering departments and manufacturing departments. These departments in their own turn have individuals that are sellers, customers, engineers and manufactures.

The association between agents and roles is done through (organizational) positions [6]. Positions that correspond to atomic roles are called atomic positions and are place holders for individual agents, while those that correspond to composite roles are called composite positions and are place holders for groups of agents. Positions are related via the “contains” relation as well. Any containment relation between positions that has not been anticipated in the roles’ layer is not allowed between positions. Positions inherit responsibilities and recipes from the corresponding roles and they can have additional ones. Each role can be associated with one or more positions.

As Fig. 1 shows, the containment relation between positions directly reflects the containment relation between roles. For example, the position “Engineer” of an individual agent that represents Mike is contained to the position “E-dept” that in is own turn is contained to the position “The Company”. The positions layer is referred to as the organization structure, while the roles layer is referred to as the organization model.

Fig. 1. Part of the organization that represents a company.
Groups of agents are denoted by the capital letter $G$. When a set of agents is assigned to a composite position, then the group plays the corresponding role. The same relation holds between individuals that fill atomic positions and the corresponding atomic roles.

3 Responsibilities

Responsibilities are normative for agents and comprise a goal state and a condition. States and conditions are atomic first-order predicate-logic formulae that may contain free variables. The application of a list of substitutions $\sigma$ to a state $s$ is denoted $s|_{\sigma}$.

A responsibility is a rule of the form $s \Rightarrow g$ where $g$ is the goal state and $s$ the corresponding condition. Since the responsibility can have variables, we put the restriction that the variables of the goal state must appear in the corresponding condition. In case a responsibility has variables, then it is a responsibility template and at a specific time point it can have several instances. For example, the responsibility template, “if there is a customer order, then orders must be scheduled” has one instantiation for each customer order known at a specific time point.

Responsibilities are assigned to roles and are either individual or group responsibilities. If $s \Rightarrow g$ is an individual responsibility of a role that an agent plays and the agent believes that an instance $s|_{\sigma}$ of $s$ holds, then it must see to it that (stit) the goal state $g|_{\sigma}$ holds (for stit see [10]). If $s \Rightarrow g$ is a group responsibility and the group accepts that an instance $s|_{\sigma}$ of $s$ holds, then it must see to it (collectively) that the goal state $g|_{\sigma}$ holds.

This paper focuses on group responsibilities. Due to information distribution and access restrictions in information sources within organizations, it is not possible each member of a group to recognize each instance of a group responsibility. State recognition recipes (r-recipes) represent specific organization policies that drive groups to accept that the conditions of responsibilities hold. For a group of agents to form an acceptance that a state holds, it is not necessary that all agents within the group believe that this state holds. For instance, the belief of even a single “trusted” agent can be enough for the group to form an acceptance.

4 R-Recipes

While individual agents use individual actions to form beliefs, group members use state recognition recipes (r-recipes) in order to form acceptances.

An r-recipe belongs to a specific role, which is called the relevant role, and is a policy for all groups that play that role (i.e. groups that occupy positions which correspond to this role). Each group that plays the relevant role is a relevant group for this recipe. As Fig. 2 shows, an r-recipe is associated to a specific state, which is called the recipe state.
The body of an r-recipe comprises elements of the form $\rho_{\text{ind}}^{s}$, where $\rho$ is an internal role, $\text{ind}$ an indicator, and $s$ is a subsidiary state of recipe’s state. The indicator is a quantifier for the players of $\rho$ and can take the value all or one, indicating all the players of $\rho$, or at least one of them, respectively. Each internal-role in the r-recipe must be contained (via the relation “contains”) to the relevant role of the recipe.

An r-recipe specifies that the members of a group $G$ shall form an acceptance about the recipe state, when sufficient number of individuals (resp. subgroups) of $G$ believe (resp. accept) the subsidiary states. In other words, state recognition recipes are specifications of policies that state which are the “authorized” individuals and subgroups in $G$ to disseminate beliefs and acceptances, and specify how the combination of subsidiary beliefs and acceptances yields to new acceptances.

Specifically and in order to reduce complexity, let us consider a simple r-recipe, whose body comprises a single element $\langle \alpha, \{\rho_{\text{ind}}^{s}\} \rangle$ and whose relevant group is $G$. The recipe specifies that the members of $G$ shall accept that $\alpha|_{\sigma}$ holds, if and only if:

- $\rho$ is a composite role, and all or at least one – depending on the value of $\text{ind}$ – of the subgroups of $G$ that play $\rho$ accept that $s|_{\sigma}$ holds.
- $\rho$ is an atomic role, and all or at least one – depending on the value of $\text{ind}$ – of the agents that play $\rho$ believe that a state $s|_{\sigma}$.

In case there is more than one element in the body of an r-recipe, then there must be a substitution for each subsidiary state and these substitutions must be composable. In this case, the accepted state results by applying the composition of these substitutions to $\alpha$.

Whether the internal role $\rho$ is atomic or composite, the players of this role “contribute” to the acceptance of $\alpha|_{\sigma}$ through the acceptance/belief of $s|_{\sigma}$. This is a “direct contribution” specified by the triple $(s|_{\sigma}, \langle \text{rec}, \rho_{\text{ind}} \rangle, \alpha|_{\sigma})$, where rec is the specific r-recipe used. Given the substitution list $\sigma$, the contribution shall be denoted $(s, \langle \text{rec}, \rho_{\text{ind}} \rangle, \alpha)|_{\sigma}$. The generic form of a contribution is defined in the section that follows.

It must be noticed that if $\rho$ is a composite role then the groups that play this internal role must have acceptances for the subsidiary state. This is done by using r-recipes of their group. In case $\rho$ is an atomic role, agents must have individual beliefs rather than acceptances.
The following is an example of an r-recipe:

\[
\langle \text{ordered}(p,c), \{
\text{customer}_{\text{one}}: \text{want-product}(p,c),
\text{seller}_{\text{all}}: \text{pending}(p,c)\}\rangle.
\]

This r-recipe belongs to the role “customer department” in our example organization and states that the members of a customer department shall accept that a specific product \(p\) for a customer \(c\) is ordered, when it is known that at least one customer believes that it wants the product \(p\) and that all sellers believe that the product order is pending. Fig. 3 shows a graphical representation of this recipe.

![Graphical representation of the recipe](image)

**Fig. 3.** An example of a state recognition recipe (r-recipe).

Further examples of r-recipes are provided in Fig. 4. As it will be shown, the combination of these recipes can drive members of the group “The company” to form acceptances about instances of the state “order-prepared\((p,c)\)”.

### 5 Forming acceptances

The acceptance of a state requires communication between group members and knowledge about the position of each agent in the organization.
In our example, let us consider the state “order-prepared\((p,c)\)” – where \(c\) and \(p\) are variables denoting a customer and a product. Towards the acceptance of one or more instances of this state by the members of the group “The Company”, each member must find r-recipes that specify how its individual beliefs can “contribute” to this acceptance, based on organization policies. Based on the contributions of company members, and according to the r-recipes of the company, one or more instances of the state “order-prepared\((p,c)\)” may be accepted by the group.

The following paragraphs formalize the generic notion of the “contribution” and specify how acceptances are formed based on the contributions of individual agents.

5.1 Contributions

As already pointed, given an r-recipe \(\text{rec} = \langle \alpha, \{\ldots, \rho_{\text{ind}}, \ldots\} \rangle\), players of the internal role \(\rho\) are considered to “contribute” to the acceptance of the state \(\alpha\). This is a “direct contribution” specified by the triple \((\text{s, } \langle \text{rec, } \rho_{\text{ind}}\rangle, \alpha)\) where \(\text{rec}\) is the r-recipe, \(\rho_{\text{ind}}\) is an element of \(\text{rec}\) and \(\sigma\) is a substitution list.

In general, a “contribution” of an individual agent to the acceptance of a state \(\alpha\) can be either a direct contribution of the form \((\text{s, } \langle \text{rec, } \rho_{\text{ind}}\rangle, \alpha)\) or a contribution \((\text{s, } \langle \text{rec}_1, \rho_{\text{ind}_1}\rangle, \text{s}_2, \ldots, \text{s}_k)\) followed by a direct contribution from \(\text{s}_k\) to \(\alpha\). Such a contribution is represented as a list \((\text{s, } \langle \text{rec}_1, \rho_{\text{ind}_1}\rangle, \text{s}_2, \ldots, \text{s}_k, \langle \text{rec}_k, \rho_{\text{ind}_k}\rangle, \text{s}_{k+1})\) where \(\text{s}_i\) directly contributes to the acceptance of \(\text{s}_{i+1}\) by means of the \(\rho_{\text{ind}_i}\) element of the r-recipe \(\text{rec}_i\) and \(\text{s}_{k+1}\) is equivalent to \(\alpha\).

Fig. 5 shows a contribution to a state \(\text{s}_k\), which is indicated as a path in a tree constructed by r-recipes (r-tree). Each node in the r-tree corresponds to a state, and each arc in the r-tree corresponds to an element in the body of an r-recipe.
5.2 Personal contributions

Each agent computes its own contributions to a state $s$ proactively without being asked to do so. This is done without constructing the whole r-tree and without considering the contributions of the others. The number of r-trees is determined by the number of r-recipes known by an agent, for each instantiation of a state $s$.

When an agent in a group $G$ searches for personal contributions to the state $s$ it selects an r-recipe of this group with recipe state $s$. Then, it considers only those elements of the r-recipe that concern roles played by groups in which the agent participates. For example, when an engineer searches for contributions for the state “order-prepared($p$, $c$)”, then it may use the recipe $rec_2$ in Fig. 4. This agent shall consider only the element that corresponds to the role “engineering department”, since engineers in our example organization participate only in groups that play this role.

Given a composite role $\rho$ that an agent plays, the agent can find its personal contributions for a state $s$ according to the following algorithm:

$$\text{Compute-personal-contributions} (s, \rho)$$

{Let contributions be the set of all contributions to $s$ computed} contributors is set to {}

for each r-recipe of $\rho$ with relevant state $s$ do
  for each element $\rho_{\text{ind}}$ in the body of rec do
    for each instance $s_{\text{ind}}$ that the agent believes that it holds
      contributions = contributions $\cup$ ($s_{\text{ind}}$, (rec; $\rho_{\text{ind}}$), $s$)$_{\sigma}$
    end for
    if $\rho_{\text{ind}}$ is a composite role played by a group of the agent
      $C$ = Compute-personal-contributions($s_{\text{ind}}$, $\rho_{\text{ind}}$)
      for each contribution ($s_1, \ldots, s_i$)$_{\sigma}$ in $C$
        contributions = contributions $\cup$ ($s_1, \ldots, s_i$, (rec; $\rho_{\text{ind}}$), $s$)$_{\sigma}$
      end for
    end if
  end for
end for (element in r-recipe)
end for (r-recipe)
return contributions
end

For instance, for an engineer to find a personal contribution to the state “order-prepared($p$, $c$)”, it must find all recipes towards this state. One of these recipes is the recipe $rec_2$ in Fig. 4. As already said, the only recipe element that is of interest for the engineer is the one represented by the leftmost arc. This element concerns the engineering department which is the only group in which the engineer participates. According to the algorithm, for this element, and since “engineering department” is a composite role, the agent must compute all its contributions to the subsidiary state “designed($p$, $c$)” recursively. Again, the agent finds all recipes towards this state. Let $rec_1$ in Fig. 4 be one of these recipes. This recipe has one element with internal role “engineer”. Since this is an atomic role, according to the algorithm, the agent finds all instances designed($p$, $c$)$_{\sigma}$ that it believes they hold. Let “designed(laptop, george)” be one such instance with $\sigma = \{p/laptop, c/george\}$. In this case the resulting personal contribution is
(designed(laptop,george), (rec₁, engineer, one),
designed(laptop,george), (rec₂, engineering-department, one),
order-prepared(laptop,george))

Given the same r-recipes, for each different instance of the state “designed(p,c)”
that the agent believes, a different contribution to the state “order-prepared(p,c)”
would be constructed. Contributions are multiplied when alternative recipes to states
exist.

5.3 Group contributions

Each agent in a group \( G \) computes its personal contributions to a state \( \alpha \). These
contributions must be combined in order a specific group to form acceptances.
Towards forming acceptances, one must gather the contributions of all members and
for each one of them, must check whether a sufficient number of group members
agree on it. In this case personal contributions become group contributions.

The combination of personal contributions is done by a server agent that is
knowledgeable about the personal contributions of the group. Each agent that belongs
to the group \( G \) must send its contributions to the server agent in the organization. This
agent combines the contributions, identifies potential acceptances and forms group
contributions. In an organization there may by several server agents that are
responsible to identify potential acceptances. For example, each group can have its
own server agent that accepts agent contributions and computes potential acceptances
only for its group. The algorithm for forming group contributions and thus “potential
acceptances” for specific positions by a server agent is the following:

Form group contribution

Given the contribution

\[ C = (s₁, \langle \text{rec}_₁, \rho^₁, \text{ind}_₁ \rangle, s₂, \ldots, s_k, \langle \text{rec}_k, \rho^k, \text{ind}_k \rangle, s_{k+1}) \]

for each position of role \( \rho^i \)

if the agent in position has identified this contribution
mark \( s_i \) as a belief of position

end for

for \( i = 2 \) to \( k \) do

Given the direct contribution \( (s_{i-1}, \langle \text{rec}_i, \rho^{i-1}, \text{ind}_i \rangle, s_i) \)

(*) for each position of role \( \rho^i \)

Compute the fraction \( F \) of agents/groups that
are in positions of \( \rho^{i-1} \) contained in position and
have \( s_i \) as a belief or potential acceptance

if \((\text{ind} = \text{all} \text{ and } F = 1) \text{ or } (\text{ind} = \text{one and } F > 0)\)
mark \( s_i \) as a potential acceptance of position

end if

end for

end for

compute the loop (*) for the role \( \rho \) and for \( i = k+1 \).

end

Let us assume that in our example the company has two customer departments “C-dept₁”, with seller positions “seller₁” and “seller₂”, and “C-dept₂” with seller positions
“seller₃” and “seller₄”.
Given the recipes $\text{rec}_4$ and $\text{rec}_2$ in Fig. 4, let us assume that “seller$_1$”, “seller$_2$” and “seller$_4$” have sent the following personal contribution

$$(\text{pending}(p_i, c_i), \langle \text{rec}_4, \text{seller}_\text{all} \rangle, \text{ordered}(p_i, c_i), \langle \text{rec}_2, \text{customer-department}_\text{one} \rangle, \text{order-prepared}(p_i, c_i))$$

where $p_i$ and $c_i$ are specific instances of $p$ and $c$ respectively. The server agent constructs the table shown in Table 1. Following the algorithm, for $i = 1$, the server identifies the positions of role “seller” and for each seller that has sent this contribution, it ascribes the belief for the state “pending($p_i, c_i$)”. These positions in Table 1 have been marked with a tick (✓) and the rest with a cross (✗).

<table>
<thead>
<tr>
<th>company:</th>
<th>order-prepared($p_i, c_i$)</th>
<th>The Company</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>customer-</td>
<td></td>
<td></td>
</tr>
<tr>
<td>department:</td>
<td>ordered($p_i, c_i$)</td>
<td>C-dept$_1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>C-dept$_2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td>seller$_4$</td>
<td>ordered($p_i, c_i$)</td>
<td>seller$_1$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td>seller$_2$</td>
</tr>
<tr>
<td></td>
<td></td>
<td>✓</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 1. A group contribution for “The Company”.

Continuing for $i = 2$, given the direct contribution

$$(\text{pending}(p_i, c_i), \langle \text{rec}_4, \text{seller}_\text{all} \rangle, \text{ordered}(p_i, c_i))$$

the server identifies the positions of the role “customer department” and for each position of this role it computes the fraction $F$ of sellers that (a) are contained in this department and (b) believe “pending($p_i, c_i$)”. The “C-dept$_1$” contains “seller$_1$” and “seller$_2$”. Therefore $F = 1$. The “C-dept$_2$” contains “seller$_1$” and “seller$_4$”. Thus $F = \frac{1}{2}$. Since the indicator in this direct contribution is “all” the state “ordered($p_i, c_i$)” is marked as potential acceptance for “C-dept$_1$”. However, this is not the case for “C-dept$_2$”.

For $i = 3$, given the direct contribution

$$(\text{ordered}(p_i, c_i), \langle \text{rec}_2, \text{customer-department}_\text{one} \rangle, \text{order-prepared}(p_i, c_i))$$

the server identifies the positions of the role “company” and for each position (in our case there is only one - “The company”) it computes the fraction $F$ of customer departments that are contained in this position and have the state “ordered($p_i, c_i$)” marked as a potential acceptance. In our case $F = \frac{1}{2}$ since one of the two customer departments have a potential acceptance for “ordered($p_i, c_i$)”. Since the indicator in this direct contribution is “one” the state “order-prepared($p_i, c_i$)” is marked as a potential acceptance for the group in the position “The company”.

Contribution: $(\text{pending}(p_i, c_i), \langle \text{rec}_4, \text{seller}_\text{all} \rangle, \text{ordered}(p_i, c_i), \langle \text{rec}_2, \text{customer-department}_\text{one} \rangle, \text{order-prepared}(p_i, c_i))$
It must be emphasized that in this stage, to form group contributions, the server agent does not need to take into account all the elements of the r-recipes used. The server agent considers only the recipe elements that have been used by the agents for the construction of their personal contributions. For instance, computing the group contributions for the state “ordered(p_i,c_i)”, the server agent does not need to take into account the contributions of customers, even if – according to the r-recipe rec_4 Fig. 4 – the contributions of both parties, sellers and customers, are required in order a customer department to form an acceptance concerning the state “ordered(p_i,c_i)”. Therefore, during the identification of group contributions the server agent forms “potential acceptances” rather than acceptances.

As Section 5.4 shows, given the r-recipe rec_4 in Fig. 4, the state “ordered(p_i,c_i)” shall be accepted by one customer department if and only if, at least one of its customers have expressed – via its personal contribution to the state “ordered(p_i,c_i)” – its belief to the state “want-product(p_i,c_i)” and all its seller agents have expressed – via their personal contribution to the state ordered(p_i,c_i) – their belief to the state “pending(p_i,c_i)”.

### 5.4 Acceptances

To form acceptances for a position, the server agent must combine the group contributions of this position, checking whether the requirements of r-recipes are satisfied. This means that for each recipe element $\rho_{ind:s}$ there is a contribution towards $s_i$ from a group or individual agent that plays the role $\rho$.

For example, for the state “order-prepared(p_i,c_i)” to be accepted, according to the recipes in Fig. 4, all group contributions shown in Fig. 6 are necessary.

<table>
<thead>
<tr>
<th>A</th>
<th>C_1</th>
<th>C_2</th>
<th>C_3</th>
<th>C_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>order-prepared(p_i,c_i)</td>
<td>order-prepared(p_i,c_i)</td>
<td>order-prepared(p_i,c_i)</td>
<td>order-prepared(p_i,c_i)</td>
<td></td>
</tr>
</tbody>
</table>
| rec_2     | engineering-department
  manufactured(p_i,c_i) | rec_2     | manufacturing-department
  manufactured(p_i,c_i) | rec_2     | customer-department
  manufactured(p_i,c_i) | rec_2     | customer-department
  manufactured(p_i,c_i) |
| designed(p_i,c_i) | designed(p_i,c_i) | designed(p_i,c_i) | designed(p_i,c_i) |

<table>
<thead>
<tr>
<th>B</th>
<th>C_1</th>
<th>C_2</th>
<th>C_3</th>
<th>C_4</th>
</tr>
</thead>
<tbody>
<tr>
<td>order-prepared(p_i,c_i)</td>
<td>order-prepared(p_i,c_i)</td>
<td>order-prepared(p_i,c_i)</td>
<td>order-prepared(p_i,c_i)</td>
<td></td>
</tr>
</tbody>
</table>
| rec_1     | engineering
  designed(p_i,c_i) | rec_1     | manufacturing
  designed(p_i,c_i) | rec_1     | customer
  designed(p_i,c_i) | rec_1     | seller
  designed(p_i,c_i) |
| designed(p_i,c_i) | designed(p_i,c_i) | designed(p_i,c_i) | designed(p_i,c_i) |

**Fig. 6.** Group contributions

The server agent must combine these contributions and identify that they satisfy the requirements of the r-recipes used. Fig. 7 shows how the set of contributions in Fig. 6 satisfy the requirements of the r-recipes in Fig. 4 for the acceptance of the state “order-prepared(p_i,c_i)”.
Given a set of group contributions (Contributions) of a specific position (Position), (the state of each contribution in this set is a potential acceptance of the group in Position) the server agent runs the following algorithm to form acceptances.

Accepted-States(Contributions, Position)
Let REC be a set of r-recipes, initially empty
while Contributions not empty
  If a subset of Contributions have a unifier σ, then apply σ to all contributions
  It is assumed that no couple of contributions share the same variable
  Let C=(S₁,〈rec₁, ρ₁ ind₁〉, S₂,…,S_k,〈rec_k, ρₖ ind_k〉, Sₖ₊₁) be in Contributions
  Let E=(S₁,〈rec₁, ρ₁ ind₁〉, S₂)
  (*) Choose a recᵰᵣₑc₁ in REC, and let u_rec₁ be that recipe. In case no such recipe exists in REC, let u_rec₁ be rec₁.
  For the r-recipe u_rec₁ do the following:
  an element in the body of a r-recipe is marked as “pending’ when there is no direct contribution
  that satisfies it
  -Let (ρ₁ ind₁:S) be a pending element of the r-recipe u_rec₁ such that
    S|τ=S₁|τ.
    In case no such element exists, then backtrack to the choice point (*)
  -Mark the element (ρ₁ ind₁:S) as “satisfied”
  -Apply τ to all the elements and to the relevant state of u_rec₁
  -Mark the rest of the elements of u_rec₁ that are not “satisfied” as
    “pending”
  -Add u_rec₁ in REC
  Backtrack to the choice point (*) [to choose another recipe in REC]
  [This is necessary for each contribution to satisfy as much pending recipe elements as possible]
Update Contributions by replacing C with
 C’=(S₂,〈rec₂, ρ₂ ind₂〉,…,Sₖ₋₁,〈recₖ₋₁, ρₖ₋₁ indₖ₋₁〉, Sₖ₊₁)
end while
for each r-recipe rec in REC whose relevant state is Sₖ₊₁|σ and has no pending elements
if Check-accepted-state(rec)
  Form the recipe state of this r-recipe as an acceptance of Position.
end for
end
Checking accepted state

\begin{algorithm}
\textbf{Check-accepted-state} \textit{(recipe)}
\begin{algorithmic}
\If{(for every element \((\rho, \text{ind}) : S\) of recipe)}
\If{S is believed by individual agents (the role \(\rho\) is atomic) or there is a recipe \(\text{rec} \in \text{REC}\) with \(\text{recipe state} \ S\) and no pending elements and check-accepted-state(\text{rec}))}
\State \text{return true}
\Else \text{return false} \EndIf
\EndIf
\end{algorithmic}
\end{algorithm}

According to the above algorithm, given the set of contributions for the group “The Company” depicted in Fig. 6 the server agent checks whether there is a subset of contributions that can be unified. Since no such subset exists, the server chooses a contribution. Let that be \(C_4\) and let \(C_4^B\) be the part B of this contribution, shown in Fig. 6. Given that \(\text{REC}\) is initially empty, the algorithm uses the recipe \(\text{rec}_4\) in Fig. 4. Choosing the corresponding (pending) element of this recipe, it computes the unifier \(\tau = \{c/c, p/p\}\). It marks this element of this recipe as “satisfied” and the other element as “pending”. It computes \(\text{rec}_4|\tau\) and adds the new recipe in \(\text{REC}\). The set of contributions is updated by replacing \(C_4\) by its part \(C_4^A\).

We must notice that according to the algorithm, the recipe \(\text{rec}_4|\tau\) does not replace \(\text{rec}_4\) in \(\text{REC}\). This allows \(\text{REC}\) to contain recipes with pending elements, which can be satisfied by contributions to multiple instances. Doing so, the server agent identifies all different instances of states that are accepted.

Although the server agent may choose any contribution in the contributions set, let us suppose that it chooses \(C_3\). Let \(C_3^B\) be the part B of this contribution, shown in Fig. 6. Given that \(\text{REC}\) contains \(\text{rec}_3\), the algorithm uses this recipe. Locating the corresponding (pending) element of this recipe, which has been instantiated according to \(\tau\), it identifies that it can be unified with \(C_3^B\). In this case the unifier is the empty set. It marks this element of the recipe as “satisfied” and adds \(\text{rec}_4|\tau\) recipe in \(\text{REC}\). The set of contributions is updated by replacing \(C_3\) by its part \(C_3^A\).

Now, the set of contributions contains a subset of contributions, i.e. \(\{C_3^A, C_4^B\}\) that can be unified. The resulting set after this unification contains \(\{C_3, C_2, C_4^A\}\).

Continuing for the rest of contributions, the server finally succeeds to conclude that the state “order-prepared(\(p_i, c_i\))” shall be accepted by “The Company”, since the recipe \(\text{rec}_2|\sigma\) in \(\text{REC}\) according to the check-accepted-state function has no pending elements, and every subsidiary state can be accepted.

6. Concluding remarks

Collaborative activity between agents is a type of complex group behavior that requires sharing of knowledge and communication to create common awareness. Models of collaborative decision making and action use the notion of collective belief to cope with group awareness. However, collective belief is neither sufficient nor efficient for organizations members to act collaboratively in distributed and dynamic settings.
This paper focuses on the specification of group policies for forming acceptances in organized settings by means of state recognition recipes. According to our knowledge, the concept of acceptance has not been used for creating awareness in multi-agent systems.

The paper proposes a specific set of constructs for building organization models and structures which is by no means complete. Future work concerns the extension of this set. Furthermore, extending the expressiveness of r-recipes is a major point of future research. For instance, we shall examine a more rich set of indicators as well as the addition of constraints or percentages in the indicators in order to be able to specify more sophisticated organization policies.

Finally, another interesting point for future work concerns the exploitation of the above structures for the specification of decision-making policies for the collaborative activity itself.

Acknowledgments

This research is supported by the Pythagoras grand no. 1349 under the Operational Program for Education and Initial Training.

References