Plan is now in the figure (an error had occurred in the labelling of one of the boxes in this diagram). In fact, Figures 1 and 2 have been revised together in order to ensure that both are compatible and with the text.

9. Better integration requested

We have done this by inserting references to CRS and CAS, defined and discussed in Section 4 so that there is a more holistic feel to the paper in toto.

10. Does section 6 mention the CRS and CAS?

Yes, indeed we mentioned the CRS and CAS at least three times except that we used them in unabbreviated forms, i.e., communication representative structures and authorization structures, respectively. In this revision, we use both the abbreviated forms (defined on first usage) and the unabbreviated forms according to the contexts where they appear.

We trust that, having responded by the deadline (October 5) to all the useful review comments, you will find the revised manuscript in a form now suitable for publication. Please let us know if there are any other modifications required, either in context of format. With many thanks

Professors Brian Henderson-Sellers and John Debenham and Dr Wenpin Jiao
Abstract: A group of organizational models for multi-agent systems are described in which an interaction pattern plays a central role. From the models we have developed, we can easily 1) identify the roles, their responsibilities and interaction protocols in which they are involved and 2) capture agents, their services and their communication capabilities. Furthermore, by relating all of those models together, we can obtain an overall view about a multi-agent system. Thus, when building multi-agent systems, developing those organizational models are the main tasks for analysis and design. These theoretical developments are illustrated with an example system for making airline bookings.
1. Motivation

Multi-agent systems are often viewed as natural metaphors of information systems in the real world (Wooldridge and Ciancarini, 2001). They are useful not only for the creation of (multi-agent) information systems but also describing the organizational goals themselves by simulating organizational behaviour.

When people investigate organizations, these organizations and their behaviours are often studied from the aspects of organizational goals, organizational structures and the entities involved in the organizations in pursuit of organizational goals. For example, in Fuxman et al. (2001), organizations are described in terms of goals, roles and dependencies among roles while roles are for achieving or maintaining the goals. However, on the other hand, multi-agent systems are usually modelled as collections of entities (i.e., roles, agents) and interactions among entities. That means, multi-agent systems, as modelled in most of the existing methodologies, provide views that sit uncomfortably with those of organizations. It would appear, therefore, that a better approach to the development of multi-agent systems for simulating the behaviours and structures of organizations should possess the capability of providing views matching with those of the organizations themselves.

Though there may be discrepancies between the descriptions of multi-agent systems and organizations, interactions are common concerns in both descriptions. Interactions are links to connect organizational entities together on the one hand and are key elements for software entities in constructing software systems on the other hand. If we can relate organizational models to multi-agent systems through interactions, we can build multi-agent systems that could simulate the behaviours of organizations in ways that are more appropriate. Furthermore, via the link of interactions, organizational views may provide us with a new approach to analyze and design multi-agent systems.

To model a multi-agent system, many issues should be addressed at the organizational level. Firstly, we must know what the system is supposed to do or what the system’s goals are, what roles are involved in the system, how the goals are achieved and tasks allocated to roles, and what agents will play each role. Secondly, we should be clear about how the agents involved in the system achieve the goals by carrying out tasks cooperatively, efficiently and economically, what resources and skills the agents should possess to perform the assigned tasks and how the agents communicate in their interaction with other agents and the environment. Thirdly, since agents are autonomous entities and some of them are self-interested, we should study what autonomous behaviours of agents can be allowed or, conversely, should be prohibited in the system and how the organizational (or social) constraints may compromise the autonomy of individual agents (Yu et al., 1997; Ossowski, 1999; Esteva et al., 2001).

In this paper, we analyze some of the above issues to model a multi-agent system organizationally from several aspects and then discuss how to draw an overall view of the system from those models.

By modelling systems in a series of organizational structures, we anticipate being able to simplify the process of analysis and design of multi-agent systems. Here, we study what process might be suitable for building multi-agent systems by using these models.
In our approach, we first model organizations from the aspect of the goals that the organization is pursuing. In goal-driven organizational models, goals are decomposed in specific structures that are compliant with specified interaction patterns that roles use to interact with one another while achieving those goals. Based on these models, we aim to identify what roles are necessarily involved in pursuing the achievements of the goals and how roles interact while taking actions.

In the goal-driven organizational model, interaction patterns only abstractly define the control relationships among roles. However, not all roles involved in organizations are required or have the right to take part in communications with others while the organizational goals are being achieved. Thus, we define some communication structures to model the situations in which some agents may communicate with entities outside the organization and others that may not.

By modelling multi-agent systems as goal-driven organizations, we can easily identify what roles may be involved in the systems and what are the interaction constraints that roles should satisfy while achieving the goals of organizations. Furthermore, by analyzing the communication structures of organizations, we can capture which agents may participate in the interactions and how those agents communicate with one another. Indeed, the goals themselves may be at a very high level of abstraction commensurate with the organization.

In the following, Section 2 defines some concepts used throughout the paper. Section 3 describes a goal-directed model for multi-agent systems. Section 4 proposes some models of interaction and communication within multi-agent systems. Section 5 combines these models into a whole in representing multi-agent systems. Section 6 suggests a process for analysis and design of multi-agent systems, in which the main tasks of each stage are to build related organizational structures and then to identify roles and agents from them. Section 7 gives an example of how to apply the process to build multi-agent systems. Finally, Section 8 offers some conclusions and directions for future research.

2. Concepts

2.1. Agent

Although there is no unanimously accepted definition for an agent, it can generally be considered as an active communicating entity that has many features, such as autonomy, pro-activity and social ability, and can provide services via interactions with other agents or the environment (e.g. Wooldridge, 2002).

2.2. Role

In multi-agent systems, the role is an abstract representation of an individual agent or a sub-system that provides specific functions or services (Ferber and Gutknecht, 1998). A role can be characterized as a collection of responsibilities, permissions and interaction patterns. Every role involved in a multi-agent system must be handled by at least one agent. Conversely, each agent can play one or more roles (Ferber et al., 2001).

2.3. Interaction pattern

In Van Dyke Parunak et al. (2003), interactions are classified and specified based on two additional primary relations of correlation and congruence; and in Fuxman et al.
(2001), social patterns are used to abstract interactions among agents. However, those interactions described in the literature can only be considered as (semi-)formal models for some specified categories of interaction protocols.

Interaction patterns, as used here, are different conceptually from interaction protocols. Interaction protocols are concerned with the concrete steps and content that roles send and receive, whereas interaction patterns are only concerned with the control relationships among roles.

In the real world, we can identify many kinds of commonly used interaction patterns (IP), such as auction, cooperation and so on. Abstractly, we can divide interaction patterns into four categories.

1. **Centralized patterns**
   In these patterns, there often exists a central controller and a group of controlees, among which the controller controls what and with whom to interact, on the other hand, the controlee can only act under the control of the controller. For example, in the pattern of “Dictate” (or “Instruction”), the Dictator instructs what others should do and the others may or may not report back results, whilst in the pattern of “Advice”, the Advisor tells others how to take actions.

   Centralized interaction patterns can be defined as follows.

   \[
   \text{Centralized IP} = \langle C, R_2, CP \rangle
   \]

   where \( C \) is the role of controller, \( R_2 \) is a set of roles, and \( C \not\in R_2 \).

2. **Parallel (or Decentralized) patterns**
   In these patterns, interactions occur among equal roles, in which no one will place controls on others. For example, in “Cooperation”, all roles involved are equal and no one can instruct others to do something via interactions.

   Decentralized interaction patterns can be defined as follows.

   \[
   \text{Decentralized IP} = \langle R, CP \rangle
   \]

   where \( R \) is the set of roles.

   For a decentralized pattern, when the roles involved can be divided into two disjoint groups and interactions only occur between those two groups, the pattern can be considered as a special case, i.e., a peer-to-peer mode. Since peer-to-peer computing has attracted so much attention in the computer community over recent years, we propose that the peer-to-peer mode should be considered as a separate interaction pattern (see below) in order to simplify the analysis of interactions among roles.

3. **Peer-peer patterns**
   In this kind of interaction pattern, there are usually only two sides involved in the interaction. For example, “Auction” is a kind of interaction occurring between the Manager and a collection of Bidders, whilst “Bargaining” is between two competing roles.

   Peer-Peer interaction patterns can be defined as follows.
Peer-Peer IP = < R1, R2, CP >

where R1 and R2 are two sets of roles, and R1 \( \cap \) R2 = \( \emptyset \). CP is the specification of a communication protocol.

4. Hybrid patterns

In many cases, interactions among roles do not exhibit a pure interaction pattern. For example, in “Coordination”, there is a coordinator controlling the overall activities whilst participants are equally involved in the activities for such cooperation.

A hybrid interaction pattern can also be defined as a combination of two or more (pure) interaction patterns. For example, suppose that an interaction pattern is composed of a centralized pattern and a parallel pattern. Then the hybrid pattern can be defined as follows.

Hybrid IP = < C, R2, CP1 > + < R3, CP2 >

where C is the controller and R3 \( \subseteq \) R2. The symbol ‘+’ indicates that the interactions among roles can be divided into two independent groups, where the first group of interactions conforms to a centralized pattern and the second group is compliant with a decentralized pattern.

Interaction patterns are independent of organizations, so roles involved in interaction patterns are conceptually different from those participating in organizational behaviours. For instance, in “Auction”, the roles are manager (or auctioneer) and bidder (or auctionee); whereas in a real multi-agent system (e.g., an e-market) that instantiates an auction procedure, the roles are seller and buyer. To differentiate roles in an organization and roles involved in an interaction pattern, we can call the latter meta-roles.

3. Goal-Directed Multi-Agent System

Organizations are generally characterized as goal-directed (Carley and Gasser, 1999). Differently from functions, goals are objectives to be achieved by the system under consideration (Fickas and Helm, 1992) and are formulated in terms of optative statements (Zave and Jackson, 1997). Goals may refer to non-functional properties and concerns as well as functional ones.

The goals of an organization are usually decomposed into subgoals as AND/OR trees (Dardenne et al., 1993; Letier and Lamsweerde, 2002), in which the relationships or dependencies among the tasks used to achieve the goals, such as pooled, sequential and reciprocal (Carley and Gasser, 1999), are implicitly defined. Those decompositions cannot indicate how tasks will be allotted, how goals will be cooperatively achieved or how the coordinator will coordinate the cooperation. For cooperative goals, AND/OR trees may be suitable to represent the structures of tasks while, for competitive goals, AND/OR trees may not be suitable.

Since goals must be achieved by roles eventually, how goals are decomposed and allotted should be compliant with how roles are being involved in the organization, e.g., the way that roles are organized in the organization.
Organizations have different structures usually because they have different management modes (Constantine and Lockwood, 1994). For instance, a limited corporation and a public university have different organizational structures because their management modes are significantly different. A huge company is structurally different from a small company because they possess different management modes. If we consider the management mode as a special kind of interaction pattern, we will find that the interaction pattern will decide the structure of an organization. In other words, roles are organized according to some special interaction patterns.

If we model a goal in a structure to be compliant with a specific interaction pattern, we can reason about not only what dependencies exist among tasks but also how those tasks will be allotted to roles and how roles will interact to achieve the goal.

3.1. Goal Decomposition Structure

For each goal, the plan to achieve it is decomposed into a set of subgoals (a.k.a. interim goals) with associated tasks (Figure 1). Those tasks are allotted to a group of roles; the roles must adapt a specific interaction pattern to be pursued in order to achieve the goal. If we know what interaction pattern is suitable for the goal in advance, we can be confident about the best way to decompose the plan into goals and tasks, how to assign those tasks to roles and how the roles work together to achieve the goal.

No matter what the interaction pattern is, a plan can be decomposed into a series of tasks and interim goals, where tasks will be handled by roles.

The general structure of plans can be formally defined as follows.

Plan = < IP, PS >

where IP is the interaction pattern to be used to achieve each (interim) goal, PS is the structure of plans, and

PS = < T >

where T is a set of tasks for that particular goal or interim goal. For any t ∈ T, t = < F, R, S >, where F is the function to implement the task, R and S are sets of resources and skills required to carry out the task.

Figure 1. General Structure of Goals, Plans, Tasks and Interaction Patterns
In this definition, the plan must be related to a special interaction pattern and any task should be performed by a role.

To carry out a task, resources and skills requirements should be satisfied eventually, i.e., the role undertaking the task should possess adequate resources and skills to perform the task.

If we now include the interaction patterns into the definition, we can refine the definition as follows.

For goals to be achieved in peer-peer interaction patterns,

$$\text{Peer-Peer PS} = < T_1, T_2 >$$

where $T_1$ and $T_2$ are two sets of tasks, $T_1 \cap T_2 = \emptyset$, and $T_1$ will assigned to roles at one peer and $T_2$ to roles at the other peer.

For goals to be achieved in centralized interaction patterns,

$$\text{Centralized PS} = < t, T_2 >$$

where $t$ is the task to be assigned to the controller, $T_2$ is a set of tasks to be allotted to controllees and $t \notin T_2$.

For goals to be achieved in decentralized interaction patterns,

$$\text{Decentralized PS} = < T >$$

where $T$ is a set of tasks to be allotted to roles.

For goals to be achieved in hybrid interaction patterns, the plan structures can be defined as combinations of more structures. For example, suppose that the hybrid interaction pattern is comprised of a peer-peer pattern and a centralized pattern. Then the hybrid structure can be defined as follows.

$$\text{Hybrid PS} = < T_1, T_2 > + < t, T_3 >$$

where $\{t\} \cup T_3 \subseteq T_2$.

3.2. Goal-directed Multi-Agent System Architecture

As noted above, a plan can be modeled as a collection of goals (and possibly subgoals). In goal structures, we connect roles with goals. Thus, roles in the system will be identified easily since all tasks for achieving goals will be attached to roles in the goal structures. Furthermore, we can construct goals in structures as being compliant with some specific interaction patterns, so that we can easily identify what interactions will occur among roles and how roles will interact with each other to achieve specific goals (Figure 2).
In the architecture depicted in Figure 2, a multi-agent system is modelled as an organization that possesses a collection of goals and associated plans. Each goal of the multi-agent system will be decomposed according to an interaction pattern that will be used to assist in achieving the goal. For each plan corresponding to a goal, there are a series of tasks to be performed for achieving the specified goal; each task will be allotted to one or more roles to perform it (cooperatively). Finally, agents or sub-multi-agent systems will implement the multi-agent system through playing the roles involved in the organization.

Then, goals, $G$, will be associated with plans as follows:

$$G ::= < P, R, A >$$

where $P$ is a plan used to achieve the goal and each plan will be executed by a set of tasks ($T$). $R$ is the set of roles involved in the system. $A$ is a mapping from Tasks to Roles to represent the tasks assignment, and $A = f: T \rightarrow R$.

For any $r \in R$, $r = < \text{Responsibilities}, \ldots >$.

We do not include interaction protocols in roles because the tasks they undertake will place constraints on their interactions. In other words, once roles take part in achieving some goals, the interaction patterns related to the goals are the patterns that the roles achieve in their interactions.

A role can be an individual or a sub-multi-agent system, i.e., a role involved in the system will be played by either an agent or another multi-agent system. Thus, multi-agent systems are recursively defined in our model.

4. Communications in Multi-Agent Systems

Though a goal-directed model for a multi-agent system stresses that interaction patterns should adopt roles to pursue the achievements of the goals, it does not specify
how agents interact with each other or what interaction protocols agents may adopt for communicate.

Since an interaction pattern determines the interactions among roles, actors that play roles should interact by following the specifications presented by the interaction pattern. However, the actors playing roles may be individual agent and sub-multi-agent systems as well. Consequently, the interaction relationships among roles cannot be mapped onto the communication relationships among agents.

In the real world, the behaviours of an organization can be observed from outside via the interactions of roles involved in the organization. However, there are also many unobservable interactions inside the organization. The existence of this kind of phenomenon is due to the observation that a role may be played by a group of agents who act as a whole (e.g., a department or division in an institution) and those agents may be involved in intra-group interactions. When we are modelling organizations as multi-agent systems, roles can be played by individual agents or sub-multi-agent systems. Thus, interactions among agents can be both inter-group and intra-group.

Nevertheless, if we ignore (or shield off) the internal interactions among agents occurring inside sub-systems, we will find that the external interactions outside sub-systems will have a structure similar to the interaction pattern possessed by roles.

By referring to the communication structures occurring in real organizations, we propose two categories of communication structures to model organizationally internal and external communications, respectively. The communication representative structures only take the inter-system communication relationships into consideration, in which sub-systems playing roles are encapsulated as a whole; whereas the communication authorization structures will deal with those intra-system communication relationships.

In a multi-agent system, communication occurs between several agents, between agents and sub-systems, and between several sub-systems (Serrano and Ossowski, 2002). If we regard individual agents as independent, agent-based sub-systems, the communication structures of the multi-agent system can be simplified.

4.1. Communication Representative Structure

According to the goal-directed structure of a multi-agent system, the system can be built up of a set of agent-based sub-systems. In each sub-system, there must be a group of agents involved. However, not all agents in the group can or should communicate externally. To simplify the communication structures among sub-systems, each group of agents will appoint some special representative agents and assign them privileges of communicating with agents in other groups. Nevertheless, a group may not have representatives at all if it authorizes its communication to other agents outside the group itself.

Communication representative structures (CRSs) are defined as graphs, in which nodes are agent-based sub-systems and edges are bi-directed communication links among representatives. Within each system, those agents or sub-systems that are unable or prohibited from communicating outside will authorize the representatives to communicate for them, as defined in the communication authorization structures described below. Recursively, representatives can be either individual agents or sub-systems.

Formally, a CRS graph can be defined as follows.
G_{crs} = \langle R, SS, E \rangle

where, SS represents sub-systems and, for each sub-system, there is a subset, R, called representatives.

Elements in E denote links for external communications among sub-systems. Any element in E must be used to link two representatives in sub-systems. In other words, non-representatives cannot be connected via edges in E, i.e., for any e ∈ E, if e = \langle a_1, a_2 \rangle, then a_1 and a_2 must be from two different sub-systems, being two representatives of those two sub-systems.

For instance, in Figure 3, Agent13 represents System1 in communicating with both System2 and System3, and Agent21 represents System2 in communicating with System1 whilst Agent22 communicates with System3.

\[ G_{cas} = \langle R, Ag, E \rangle \]

where Ag are agents involved in the system, R are representatives in Ag (i.e., R ⊆ Ag), and E are authorization relationships among agents.

However, the semantics of edges in E is different from that in a CRS graph. In a CAS graph, edges represent the authorization relationships among agents. In principle, for any agent that cannot communicate with the outside world, it must authorize its communications to others and, once an agent authorizes its communications out, the authorized communications must eventually be able to reach a representative agent. Thus, the authorization is reasonable and useful.
Formally, for any agent, if there is an edge fanning out from the agent, then there must be at least a path (the length of the path is bigger than 0) from the agent to a representative of the system.

I.e., \( \forall e \in E \ (e = \langle a_1, a_2 \rangle \rightarrow \exists a : \text{Ag} (a \in R \land \ (a_2 = a \lor \exists e_2, e_3, \ldots, e_n (e_i = \langle a_i, a_{i+1} \rangle, \ldots, e_n = \langle e_n, a \rangle)))) \) \( (2 \leq i < n) \)

![Figure 4. Example of Communication Authorization Structure](image)

For instance, in the system shown in Figure 4, Agent1 authorizes its communication to Agent3 whilst sub-system2 authorizes its communication to Agent3 and Agent5. Agent4 and Agent5 are representatives of this system. Agent3 is not a representative though it has some authorized communication capabilities.

### 4.3. Interaction Protocol

Interaction patterns are abstractions of interaction relationships among agents. On the other hand, CRSs and CASs define who can participate in communications among sub-systems (i.e., inter-agent communications), but they do not indicate how agents communicate and what agents say during communications.

In contrast, interaction protocols are used to specify concretely the communicating steps and the content transmitted between participants. However, interaction protocols must be consistent with interaction patterns since interaction patterns also imply the relationships of roles (or agents) involved in interaction protocols. As interaction patterns are divided into four categories, interaction protocols can also be defined into four categories, i.e., peer-peer, centralized, parallel and hybrid interaction protocols.

Similarly to the definition of interaction patterns, interaction protocols can be generally defined as follows.

General Interaction Protocol = \(< I, R, M, G(R, M) >\)

Here, \( R \) is the set of roles involved in the protocol, \( I \) is the initiator of the protocol where \( I \in R, M \) is the sequence of messages transmitted between roles and \( G(R, M) \) is a directed graph representing the communicating process in which nodes, \( V(G) \), are roles and edges, \( E(G) \), are labelled by messages.
Any interaction protocol should be an embodiment or realization of a specific interaction pattern. Therefore, for any interaction protocol and its realized interaction pattern, there must be a one-to-one mapping between roles of them.

Concretely, interaction protocols with different types can be defined by refining the above definition.

Peer-Peer Interaction Protocol = \(<I, R_1, R_2, M, G(R_1 \cup R_2, M) >\)

![Figure 5. Peer-Peer Interaction Protocol](image)

Here, as shown in Figure 5, \(R_1\) are roles belonging to one peer, \(I \in R_1\), \(R_2\) are the other peer roles, and \(R_1 \cap R_2 = \emptyset\). For any edge \(<s, r> \in E(G)\), \(\{s, r\} \not\subset R_1 \land \{s, r\} \not\subset R_2\), i.e., \(s\) and \(r\) do not belong to the same peer.

Centralized Interaction Protocol = \(<I, C, R_2, M, G(\{C\} \cup R_2, M) >\)

![Figure 6. Centralized Interaction Protocol](image)

Here, as shown in Figure 6, \(C\) is the central controller, \(I = C\), \(R_2\) are controllee roles, and \(C \not\in R_2\). For any edge \(<s, r> \in E(G)\), \(C \in \{s, r\}\) but \(s \neq r\), i.e., either \(s\) or \(r\) is the controller but not both, and controllee roles do not communicate with each other.
Parallel Interaction Protocol = < I, R, M, G(R, M) >

Here, as shown in Figure 7, I ∈ R. In contrast to the two previous categories of interaction protocols, roles involved in this kind of protocol are not divided into two groups and messages can be sent to or received from any roles.

Hybrid Interaction Protocols (Figure 8) can be defined as a combination of a pure specific interaction protocol and a general interaction protocol. For example, suppose that an interaction protocol is combined with a peer-peer sub-protocol and another kind of sub-protocol and the latter sub-protocol is running within one peer, we can define the protocol as < I, R1, R2, M1, G1(R1 ∪ R2, M1) > + < I’, R2, M2, G2(R2, M2) >. The following figure shows a hybrid interaction protocol combined with a centralized protocol and a parallel protocol, which can be defined as < I, C, R2, M1, G1(\{C\} ∪ R2, M1) > + < I’, R2, M2, G2(R2, M2) >.
5. Binding Interaction to Goal-Directed Multi-Agent Systems

Goal-directed models tell us what multi-agent systems are supposed to do and how tasks are dispatched to roles but they do not indicate how roles interact and how agents communicate in order to achieve the goals and perform the tasks. This section will study how to bind goals, interaction protocols and communications together to form a complete view of a multi-agent system.

We have constructed goals with special structures compliant with specific interaction patterns. In addition, interaction patterns determine the structures of interaction protocols. Thus, once the tasks needed to achieve a goal are allotted to a group of roles, the interaction protocol that the roles are to adapt must be consistent with the goal’s decomposition structure. Furthermore, once agents (or sub-systems) play those roles, the multi-agent system can naturally use a communication structure similarly to the structure of the interaction protocol.

For example, suppose that the multi-agent system has a goal that will be achieved by roles in a Centralized interaction pattern. Then the plan to achieve that goal can be decomposed into a collection of tasks, in which one will be assigned to the controller and others will be allotted to the controllees. Naturally, roles undertaking those tasks should adapt a special centralized interaction protocol. Furthermore, when sub-systems are playing those roles, the communication representative structure among them must be compatible with the interaction protocol; otherwise, the multi-agent system will be unable to implement the interaction protocol. The relationships among goal-directed model, interaction protocol and communication models can be shown in a layer model (Figure 9).
In Figure 9, the goal is to be achieved in a pure interaction pattern, i.e., Centralized interaction pattern. Role1 is the controller and is assigned Task1, and Role1, Role2, and Role3 are interacting with a centralized interaction protocol. To implement the interaction protocol that roles are using, sub-system1, sub-system2 and sub-system3 have appointed their own representative agents and built communication links required by the interaction protocol.

From the above example, we can note that 1) the interaction relationship among roles is compliant with the interaction pattern used to achieve a goal; and then 2) since the interaction patterns determine the control structures of communications among agents, the interaction protocol adopted by representative agents possesses a compatible structure with the interaction pattern.

6. Process for Analysis and Design of Multi-Agent Systems

To analyze and design multi-agent systems, identifying roles involved in the systems, responsibilities provided by roles and interactions among roles are some of the most important things to deal with.

Currently, many agent-oriented software methodologies have been proposed, such as the AAI Methodology (Rao and Georgeff, 1995), Gaia (Wooldridge et al., 2000) and DESIRE (Dunin-Keplicz and Treur, 1995). For example, in Gaia, the tasks at the analysis stage include:

1. Identify the roles in the system, such as individuals, departments, and organizations.
2. Identify interaction protocols associated with each role, i.e., document what protocols each role may be involved in or participate in.

3. Based on the interaction protocols, identify roles missed at the first step and further capture their responsibilities and permissions.

The tasks at the design stage include:

1. Create an agent model by aggregating roles into agent types and refine them to form an agent type hierarchy and then document the instances of each agent type by instantiating the agent model.

2. Develop a services model by concretizing properties of roles.

3. Build an acquaintance model from those interaction protocols identified at the first stage and the agent model created at the current stage.

However, there are many problems unaddressed in this methodology.

Firstly, the methodology does not mention what goals the system has or what the system is supposed to do. Secondly, at the analysis stage, Gaia requests the analyst to identify all roles involved in an organization, but it does not express why we need those roles. In fact, not all roles are necessary to contribute to achieving the system goals because the system may not be supposed to do everything that the organization is doing.

Thirdly, since the process of identifying roles is not related to the system goals, not all responsibilities identified may contribute to the system goals.

Fourthly, interaction protocols are a subsidiary property of roles. If we do not relate interactions with the way that goals are being achieved, we cannot know clearly what interaction protocols a role should possess. Thus, it is hard to identify all necessary protocols associated with roles on the one hand. On the other hand, a role may participate in many kinds of protocols, but not all interaction protocols will be used while carrying out tasks. That is to say, it is not necessary to identify all interaction protocols.

Fifthly, at the design stage, agents are aggregated from roles. Thus, agents may possess some unnecessary communication capabilities and provide services unrelated to achieving the system goals.

In Section 5, when describing ways to bind interactions to the goal-directed model of a multi-agent system, we noted that we could relate goals, roles, interactions and agents together in a group of inter-related models. In our work, the process of analysis and design of multi-agent systems starts from 1) identifying goals of the systems and then uses those models defined above to 2) identify roles, capabilities of roles, interaction protocols that the roles will participate in, 3) agents to play roles, communications among agents, functions (or services) that an agent provide, and resources and skills that agents should possess to carry out tasks. This process is illustrated in Figure 10.
In Figure 10, we can see how organizational models built for multi-agent systems are used in different phases of the development of a multi-agent system. In addition, we can see the relationships among those models in the process of developing a multi-agent system.

In the requirements phase, the main tasks are to create a goal-directed organizational model and analyze the interaction patterns used to achieve the organizational goals. By decomposing goals further, we can primarily determine the roles that will take part in the achievement of the organizational goals and their interaction modes, including the interaction protocols adopted by those roles in interactions.

Then in the analysis phase, goal-directed software architectures of multi-agent systems can be derived from the goal-directed organizational models built previously.

Next, the main tasks in the phase of multi-agent system design are to determine what agents or sub-systems will play those roles identified in the previous phases and how agents or sub-systems provide services to perform tasks undertaken by roles and how agents interact to achieve the organizational goals while carrying out these tasks. At this stage, the communication relationships among agents lie on those organizational communication structures (e.g., CRSs and CASs), for instance whether an agent can communicate with others and how an agent uses others’ communication capabilities for implementing its own communications should not violate those communication structures.

In the first stage, i.e., the requirements and planning stage, the analysis tasks include:

1. Identify goals of the system.
Generally, within the requirement statements of a software system, it must declare what goals the system should achieve. Obviously, those goals declared in the requirement statements often include both functional and non-functional goals. By analyzing the requirement statements, we can obtain the goals of the system.

2. Analyze goals and determine what interaction patterns are suitable for achieving the goals.

The interaction patterns we have defined are greatly different from concrete interaction protocols or social patterns proposed (Fuxman et al., 2001). In our definition, interaction patterns actually specify the ways of solving goals (or problems), for instance in a centralized way or a distributed way, which are more abstract than interaction protocols or social patterns. So it is quite easy to determine the interaction pattern for each goal according to the strategy for achieving the goal and the possible runtime environment of the multi-agent system.

In fact, since multi-agent systems achieve the goals of organizations via simulating the behaviours of real organizations, we can refer to the ways that real organizations interact to pursue the achievements of their goals and then determine the interaction patterns that multi-agent systems will adopt to achieve these goals. For example, in an organization with an hierarchical management mode, roles between two adjacent layers often use a centralized interaction pattern to achieve the organizational goals. In a software system based on the client/server architecture, a peer-to-peer or centralized interaction pattern is generally adopted to achieve the system’s goals.

3. Build goal structures and establish a goal-directed model for the multi-agent system.

Once the goals of the multi-agent system and the interaction patterns used for achieving the goals are determined, the goals’ decomposition structures can be established. The process of building a goal’s decomposition structure can be first planning the goal, next identifying tasks to be performed for the goal’s achievement and then analyzing resources and skills required to perform the tasks.

Actually, this stage is to build goal-directed models for the multi-agent system.

In the second stage, analysis, the main targets are to identify roles and document their properties.

   1. Find out what roles are required to achieve the system goals.

The goal decomposition structures based on interaction patterns have highlighted what tasks need to be performed to achieve goals. If we assume that each task will be undertaken by a role, we will easily be able to find out what roles are required to participate in the achievements of goals of the multi-agent system. Even if a task cannot be taken on by one role, we can consider that the task is to be carried out for achieving a non-atomic sub-goal, that is to say, the sub-goal can be decomposed further. Thus, if we decompose and plan the sub-goal until all tasks occurring in the plan can be carried out
by a single role, we can determine what roles are necessary for achieving the system goals by checking what atomic tasks should be performed.

Obviously, if a role in an organization does not take charge of any tasks, the role is not necessary to be included in the multi-agent system.

2. Identify roles.

Once the roles that are necessary for achieving the goals of the multi-agent system are determined, it will not be hard to identify the roles one by one. The next steps are then to capture all identified roles’ properties, such as what responsibilities the roles possess and what kind of interaction patterns roles use to achieve goals.

3. Based on the goal-directed system model, identify the responsibilities of roles, or what tasks the roles should undertake.

Since the goal-directed system model specifies roles and their tasks, the responsibilities of roles are to perform those tasks of which they are in charge.

4. By referring to the interaction patterns, analyze interaction protocols that roles should participate in.

To carry out a task that a role undertakes, the role may or may not interact with other roles. No matter how a role interacts with others, the interaction relationships between the role and others must be compliant with the interaction patterns while the role is performing tasks to pursue the achievements of (sub-)goals. Usually, if roles have to cooperate and interact while achieving a goal, all of them should adopt the same interaction protocol.

By analyzing the goal-directed models built at the previous stage, we can determine what roles we need to participate in the system activities and what properties they should have. By creating interaction protocol models, we can identify how those roles are to interact in order to achieve the system goals.

At the third stage, design, the main targets are to identify agents and define communication links among them.

1. Identify who are to play the roles identified at the previous stage.

To identify agents that will play a role, we can start from two aspects. From the viewpoint of existing agents or sub-systems, we can investigate whether individual existing agents or sub-systems can implement the tasks undertaken by the role and can interact or communicate in a way specified by the interaction protocol of the role. We can thus discover existing agents or sub-systems that can play the role. In this case, existing agents may be self-interested and their goals may be inconsistent or even conflict with goals of the system. Nevertheless, once an agent makes the decision to play a role, it must make a tradeoff between its own goals and the overall organizational goals and, rationally, it should attach more importance to the achievements of the organizational
goals. On the other hand, if there is no such an existing agent or sub-system, new agents or sub-systems should be developed to provide corresponding services to commit to the responsibilities assigned to roles.

2. If it is an individual agent, then develop its services and resources and skills requirements according to the tasks it will carry out, and further develop its communication capabilities required for special interaction protocols.

When an agent is developed to play a specific role, the agent should be able to provide adequate services for the role to perform the tasks to be undertaken.

In general, we do not care how individual agents provide their services, i.e., the internal implementations of agents, while developing multi-agent systems. What we care more about is whether agents can provide appropriate services for roles to achieve the goals of the systems. Nevertheless, it is not to say that agents can take actions as they wish while providing services. Since organizations will restrict the behaviours of the roles involved, agents playing the organizational roles should be restricted under the same organizational constraints. For example, while roles are performing tasks to achieve organizational goals, they should use specified interaction protocols to interact with other roles. Thus, when an agent is playing a role, the agent should interact with specific agents in a specific way following an interaction protocol specified by the role. As another example, for a role to carry out its undertaken tasks correctly, the role should possess sufficient resources and skills. Naturally, the behaviours of the role will be affected by any resource limitations and lack of skills. When agents provide services, they should consider these constraints too in order not to waste resources too much or choose selfishly a forbidden way to provide services.

3. If it is a sub-system, then consider the tasks it will undertake as its goals and re-start the process recursively. After finishing the design of the sub-system, appoint appropriate agents as communication representatives of the sub-system according to the CRSs and assign their communication capabilities required by interaction protocols.

In a sub-system, agents cooperate to perform the tasks undertaken by a role. Since not all agents within a sub-system are externally visible, agents that cannot or will not communicate with the outside should authorize their communications to those representative agents of the system according to the CASs.

By analyzing the communication structures (i.e., CRSs and CASs) defined for sub-systems and among sub-systems, we can obtain the acquaintances of every agents involved in the sub-systems.

According to those interaction protocols identified at the previous stage, we can develop CRSs and CASs among agents and sub-systems. Then we can determine which agents should participate in communications in the system and what communication capabilities they should have. By investigating the tasks that agents are to perform, we can identify what resources and skills agents should possess in order to carry out the tasks.
7. An Example System: Air Flight Booking System

An air flight booking system generally can provide services such as airline and flight inquiries and ticket reservations. If we consider the system as a goal-directed multi-agent system and we assume that all roles involved in the system will be played by software agents, the goals of the system will include airline inquiry and ticket reservation.

For the airline inquiry, the interaction takes place between customers and the system and can be modelled in a peer-peer interaction pattern. The plan to achieve a goal can be decomposed into three tasks, 1) “Submit Query” to query about flights between two cities, 2) “Process Query” to interpret the submitted query and search the airline database, and 3) “Return Results” to display the query results (Figure 11).

For ticket reservations, the interaction can also be modelled in a peer-peer interaction pattern. This goal is also decomposed into three tasks, 1) “Request” to ask the system reserve a seat after selecting a flight, 2) “Seat Available Query” to query whether there is a vacant seat, and 3) “Reserve” to reserve a seat as requested if there is an available seat (Figure 12).
7.1. Modeling the system

7.1.1. Goal-Directed Model

For those tasks mentioned above, some of them can be allotted to individual roles to carry out whereas the others cannot. For example, “Submit Query” and “Request” can be handled by a single role (e.g., a customer), and “Process Query” and “Return Results” can be coped with by a travel agency. However, to query available seats and make a reservation, the travel agency has to consult the airline company, so “Seats Available Query” and “Reserve” cannot be tackled by the travel agency independently and they should be considered as a goal of a sub-system.

Then the goal-directed model of the system (Figure 13) can be constructed from the above goal structures.

![Figure 13. Goal-directed structure of the Air Flight Booking System](image)

7.1.2. Interaction Protocols

There are two interaction protocols in the system, one for airline inquiries and the other for ticket reservations. Both are peer-peer interaction protocols.

For the protocol for airline inquiries, the two peers are the customer and the travel agency (Figure 14). The customer first sends a query about the airline information and then the travel agency returns the result back to the customer.

![Figure 14. Interaction protocol for Airline Inquiry](image)

For the protocol for ticket reservations (Figure 15), the two peers are also the customer and the travel agency. The customer sends a booking request to the travel agency and then the travel agency confirms to the customer whether the booking has been successful or not.
7.1.3. Communication Representative Structure

In the goal-directed model, the tasks “Seats Available Query” and “Reserve” are assigned to two sub-systems. Within those two sub-systems, the travel agency is the representative of both of them. The communication representative structure for the system is shown in Figure 16.

![Figure 16. Communication Representative Structure in the Flight Booking System](image)

7.2. Analysis and Design of the System

7.2.1. Roles and their Responsibilities

>From the goal-directed model of the system, we can easily identify four roles: one is the travel agency, one is the customer and the others are two sub-systems. The travel agency is responsible for processing queries and returning query results to the customer. The customer is responsible for submitting queries for airline information and requests for ticket reservations. The two sub-systems are in charge of querying available seats and reserving tickets.

If we analyze the two sub-systems further, we can identify another role involved in the system, i.e., the airline company, which will provide information both on the availability of seats and on ticketing.

7.2.2. Interaction Protocols

Those interaction protocol models shown in the previous sub-section have clear implications for what interaction protocols each role is associated. For instance, the customer participates in both the protocol for airline inquiries and the protocol for ticket reservations. So does the travel agency.
7.2.3. Agents

The roles of travel agency and customer can be played by any agents that can provide services such as “Submit Query”, “Process Query”, and so on.

Suppose that there is a Web server agent playing the travel agency role. Then services that the server agent will provide include “Process Query”, “Return Results”, “Seats Available Query”, and “Reserve”, among which the latter two are cooperatively provided by airline companies. For those agents, they must possess communication capabilities to interact with their peer agents. For instance, the agent playing the customer role must be able to communicate with the agent playing the travel agency role.

8. Conclusions

In this paper, we have described a group of organizational models for multi-agent systems. From those models, we can easily 1) identify the roles to be involved in the system, 2) determine the responsibilities of roles and 3) clearly identify the interaction patterns among roles when they are achieving goals. Furthermore, by relating all of those models together, we can provide an overall view about a multi-agent system, in which agents, their provided services and their communication capabilities can be easily identified and captured. Thus, when building multi-agent systems, developing those organizational models are the main tasks of analysis and design.

Currently, our approach may be more suitable for developing multi-agent systems that can simulate the behaviors of existing organizations in which the organizational goals and organizational structures are knowable in advance. In fact, if an organization is formed by a group of agents dynamically and the organizational goals are dynamically formed by the coalition of agents, it is almost impossible to analyze and design the multi-agent system to simulate the dynamic organization. Therefore, when developing a multi-agent system, we have to make some assumptions and, when analyzing the MAS, we can only point out what properties a role or an agent should possess and assume that an agent that commits to certain tasks should eventually perform these tasks. However, this paper does not and cannot cover all issues occurring in multi-agent systems. This is a topic for future research. In particular, as we have pointed out, the interaction patterns determine the structures of goals and what roles will be involved in the systems and, furthermore, how those roles interact to achieve goals. Once a goal is to be solved in another pattern, its structure will also change. Consequently, the number of roles and the interaction protocols may change correspondingly, as could the structure of the system as well. Thus, modelling these changes in advance may provide us with more stable system architectures. In our future research, we aim to study how to model dynamic goal structures and bind those dynamics into relatively stable systems. Secondly, the design process presented here is overly simplistic in that we do not take into consideration the autonomy of agents or the constraints of organizations or the environment. How to consider these factors in a design process and how these factors may affect the design process are questions to be addressed in future work.

Acknowledgements

WJ wishes to acknowledge support from the National Natural Science Foundation of China (grant number 60303004) and the National Grand Fundamental Research 973
Program of China (grant number 2002CB31200003). JD and BH-S wish to acknowledge financial support from the University of Technology, Sydney. This is contribution number 04/31 of the Centre for Object Technology Applications and Research.

References


