

Allocating Goals to Agent Roles during MAS Requirements Engineering

Ivan Jureta¹, Stéphane Faulkner¹, and Pierre-Yves Schobbens²

¹ Information Management Research Unit, University of Namur,
8 Rempart de la Vierge, B-5000 Namur, Belgium
iju@info.fundp.ac.be, stephane.faulkner@fundp.ac.be

² Department of Informatics, University of Namur,
8 Rempart de la Vierge, B-5000 Namur, Belgium
pys@info.fundp.ac.be

Abstract. Allocation of goal responsibilities to agent roles in Multi-Agent Systems (MAS) influences the degree to which these systems satisfy nonfunctional requirements. This paper proposes a systematic approach that starts from nonfunctional requirements identification and moves towards agent role definition guided by the degree of nonfunctional requirements satisfaction. The approach relies on *goal-dependencies* to allow potential MAS vulnerabilities to be studied. In contrast to related work where organizational patterns are imposed on MAS, roles are constructed first, allowing MAS organizational structures to emerge from role definitions.

1 Introduction

Requirements engineering is concerned with the identification of goals to be achieved by an information system (IS), the operationalization of these into the specification of IS services and constraints, the identification of resources required to perform those services, the assignment of responsibilities for the resulting requirements to agents, such as humans, devices and software.

At an abstract level, MAS are conceptualized as organizations of autonomous, collaborative, and goal-driven software components [32]. Flexibility, modularity, and robustness are some of the qualities hoped from MAS, making them an attractive choice for a range of applications, such as peer-to-peer computing, electronic commerce, etc.

There is widespread agreement that nonfunctional requirements need to be considered early in any IS development process (e.g., [9], [6], [26], [19]) in order to assist reasoning about alternative system structures. While various approaches have been proposed to transform nonfunctional requirements into functional system characteristics during system development (e.g., [21], [11], [16], [22], [7]), the specific issue of using nonfunctional requirements to allocate goals to agent roles during the RE step of MAS development has received limited attention, and no systematic approach has been proposed. In this context, this paper proposes a preliminary approach to allocat-

ing goals to agent roles. It is situated within the RE step of MAS development process and builds on widely accepted techniques for Goal-Oriented RE (GORE). The *organizational* MAS engineering metaphor is adopted, leading to the allocation of goals to roles instead of agents, in order to allow encapsulation and modularity non-functional requirements to be adequately addressed [32].

The proposed goal allocation approach advances the state of the art in three ways: (i) A systematic approach that starts with the identification of nonfunctional requirements and progressively moves towards the generation of and selection between alternative MAS role structures is proposed. It allows the choice of goal allocation to agent roles to be justified in relation to the identified nonfunctional requirements. (ii) A novel type of dependency relationship between goals is used to support the generation of, and selection between, alternative MAS role structures. (iii) Heuristics for generating and selecting alternative goal-to-role allocations are proposed.

Because the first step of the approach reuses the accepted goal analysis techniques, there are no obstacles to integrating it into existing MAS Goal-Oriented Requirements Engineering (GORE) frameworks.

Related Work. Within the RE field, GORE frameworks (e.g., [26], [28]) have been shown as useful when engineering MAS requirements (e.g., [4] and related).

In GORE research, the NFR framework [21], [6] has been the first to propose a representation language for nonfunctional requirements and to suggest a method for relating them to functional requirements. It has been adapted in Tropos, an agent-oriented development methodology [4], to deal with nonfunctional requirements in agent systems. In Tropos, goals are allocated to roles or agents through dependency links that indicate the need of an agent (the depender) to collaborate with another agent (the dependee) in order to achieve a goal, to have a task executed, or a resource provided. In [4], the selection of a MAS role structure consists in instantiating one of the predefined patterns of organizational structure (such as structure-in-five, pyramid, joint-venture, etc.). Using the qualitative reasoning techniques from the NFR framework, a pattern is selected by comparing the degree to which each alternative pattern satisfies the identified MAS nonfunctional requirements. The roles and their interdependencies are thus predefined (i.e., an organizational structure is selected and roles in that structure are instantiated), whereas the approach proposed here constructs roles first, allowing the organizational structure to appear from role definitions. In this respect, the latter approach seems less rigid for tailoring the MAS structure to nonfunctional requirements. It is guided by the goal dependency relationship to help the engineer to allocate interdependent goal pairs to roles so as to internalize or externalize the dependencies.

The MaSE methodology [29] supports security nonfunctional requirements at RE time by identifying negative use cases. The RE step (analysis phase) of the methodology involves goal identification, use case generation for goal achievement, and agent role definition. Goal allocation to roles is not treated in a systematic manner and no heuristics are provided. Similar remarks are relevant for the MESSAGE [3] MAS development methodology.

Our goal-to-role allocation approach complements the techniques discussed here. Our approach can be combined to [4] to introduce an additional technique to generate

and choose between alternative agent roles, while relying on a formal nonfunctional requirements representation.

2 A Process for Allocating Goals to Agent Roles

The proposed goal to role allocation approach consists of three steps: (i) create a consistent goal tree containing precise requirements; (ii) identify goal dependencies; (iii) generate and select between alternatives goal-to-role allocations. Because the first step reuses the well-known goal analysis techniques [8, 29] and due to a lack of space, it is not presented in this paper. Application of the suggested techniques is illustrated with examples of peer-to-peer MAS requirements discussed informally in [12].

2.1 Identify Goal Dependencies

This section first overviews the dependency conceptualization commonly used in the MAS RE literature. The possibility of employing another useful dependency type at the MAS RE step is discussed. Finally, tactics for finding and checking the completeness of the identified, so-called *goal-dependencies* are proposed.

Dependencies between Agents. In MAS RE, the following definition of a dependency relationship has been adopted in i* [31], Tropos [4], GRL [20], and REF [10]:

“A dependency link is a directed link that goes from the depender to the dependee; it can connect an agent to a hard or soft goal, a task, a resource, or vice-versa. In particular, an agent is linked to a goal, a task or a resource when it depends, in some way, on that goal to be achieved, that task to be performed or that resource to be provided; a goal/task/resource is linked to an agent when it depends on that agent to be achieved/performed/provided.” [10]

The goal-to-role allocation approach presented in this paper relies on another type of dependency: *goal-dependencies*. Goal-dependencies are studied *before* knowing which agents will be responsible for goal achievement, whereas agent-dependencies are identified from system agent intentions *after* the agents are known. While agent-dependencies allow easier characterization of *existing* (e.g., organizational) conditions, the goal-dependencies are used to define *new* roles.

Dependencies between Goals. As discussed in [26], many goal link types have been proposed to relate goals with each other, or with other elements of the requirements models: (i) *Refinement links* of two kinds have been suggested. *AND-refinements* relate a goal to a set of sub-goals (the set if called a goal refinement), meaning that achieving all sub-goals in the refinement is sufficient for satisfying the parent (or refined) goal. *OR-refinement* links a goal to alternative refinements. The achievement one of the refinements is sufficient to satisfy the refined goal. (ii) In NFR [21], weaker versions of refinement links relate nonfunctional goals and functional goals. The notion of goal satisficing has been introduced, and *contribution links* express that sub-goals are expected to contribute to the parent goal.

After constructing a refinement of a goal, the requirements engineer knows that the achievement of the sub-goals is sufficient for the refined goal to be achieved. But

refinement links do not indicate the *sequence* in which the sub-goals need to be achieved in order for the parent goal to be achieved. This information is encoded in temporal logic in the KAOS [18] and Formal Tropos [13] frameworks. We believe it is worth making it more explicit, since it will allow specific tools and techniques to be used, and it may be easier to identify sub-goals by considering the sequence of activities for (parent) goal achievement [23].

We propose thus a new type of inter-goal relationship, named *goal-dependency*. Formally, a goal g_2 depends on g_1 when:

$$\neg g_1 \Rightarrow \neg g_2 \mathbf{W} g_1$$

The classical temporal operator \mathbf{W} is read “unless” (e.g., [19]), and means that the condition on its left stays true unless the condition on its right becomes true. Intuitively, this condition means that g_2 cannot be achieved unless g_1 is achieved.

Identification of Goal-Dependencies. Consider the two goals below. The first formalizes the condition for a file transfer to be started in the case-study P2P application, the second the condition for a requested file to be considered as found.

Goal: RequestedFileFound

Definition: When the file has been requested and at least one peer p_x having and sharing the file is found, the peer p_1 that requested the file knows that the file is found, and the routing peers are found.

FormalDef: $\forall p_1: \text{Peer}; \text{fl}: \text{File}; \text{rID}: \text{RequestID}; p_1.\text{req}=\text{fl} \wedge (\exists p_x: \text{Peer}; p_x \in P \wedge p_1 \neq p_x \wedge \text{fl} \in p_x.\text{file_list} \wedge \text{Share}(p_x, \text{fl}, \text{rID})) \Rightarrow \text{Found}(\text{fl}, \text{rID})$

Goal: RandomPeersForDataRoutingFound

Definition: The n random peers for data routing are found when n routing peers are found and each peer confirms availability.

FormalDef: $\forall p_s, p_r: \text{Peer}; \text{fl}: \text{File}; \text{rID}: \text{RequestID}; \text{Found}(\text{fl}, \text{rID}) \wedge \text{Sender}(p_s, \text{fl}, \text{rID}) \wedge \text{Receiver}(p_r, \text{fl}, \text{rID}) \wedge @(\exists p_1, \dots, p_n: \text{Peer}; \{p_s, p_r\} \cap \{p_1, \dots, p_n\} = \emptyset \wedge \diamond_{\leq 5s} \text{Available}(p_1, \text{rID}) \wedge \dots \wedge \diamond_{\leq 5s} \text{Available}(p_n, \text{rID})) \Rightarrow \diamond \text{RoutingPeers} = \{p_1, \dots, p_n\}$

The existence of predicates that constrain values of the same MAS properties within different goals indicates that there may be a goal-dependency between the two goals. In the example, the property constrained in both goals concerns MAS behavior related to file transfers. It is a goal-dependency, since they have to be executed in this sequence. Notice that the refinement relationships cannot be used to determine the sequence between these goals as they are brother sub-goals.

Applying the reasoning described above, it can be seen that the property $\text{Found}(\text{fl}, \text{rID})$ appears in both $\text{RandomPeersForDataRoutingFound}$ and $\text{RequestedFileFound}$ goals’ specifications. Domain/solution knowledge allows affirming that a file first needs to be found before searching for random peers that will be used to route the file. It is thus assumed that there is a goal-dependency in which the achievement of $\text{RandomPeersForDataRoutingFound}$ depends on the achievement of $\text{RequestedFileFound}$ goals. To accept the reasoning above allows a goal-dependency identification technique to be proposed:

- (I1) If there is at least one MAS property, constrained in predicates that occur in formal specifications of two goals g_1 and g_2 , then there is a goal-dependency between them. The direction of this goal-dependency is undetermined.
- (I2) If temporal operators in formal specifications of goals in a goal-dependency make it possible to establish the sequence of achievement of one in relation to the

other goal, then the goal-dependency relationship between them is directed from the goal whose achievement precedes the other goal's achievement.

- (I3) If the goal-dependency direction cannot be determined using (I2), then domain/solution knowledge can be used to make an assumption and choose the goal in the goal-dependency whose achievement precedes that of the other goal.

Information about the temporal sequence of achievement of goals involved in the goal-dependency can only be extracted from explicit temporal operators of the predicate which constrain the MAS property giving rise to a goal-dependency. If there are no temporal operators associated with at least one of the predicates that constrain the relevant property, domain/solution knowledge will be the foundation for determining the goal-dependency direction. If neither (I2) nor (I3) allow the goal-dependency direction to be determined, then the direction remains undetermined. In the example above, there are few temporal operators in the goal specifications that allow (I2) to be useful. However, experience and the wide use of P2P applications allow the requirements engineer to make a reasonable assumption that a requested file first needs to be found before searching for routing peers (hence, (I3) is used).

The goal-dependency in the example can be specified with:

Goal-Dependency: FindFileThenSearchForRoutingPeers
Definition: Search for routing peers after the file to be transferred is found.
Involves: RequestedFileFound, RandomPeersForDataRoutingFound.
Direction: RandomPeersForDataRoutingFound **DependsOn** RequestedFileFound.
CommonProperties: Found(fl,rID)

The proposed goal-dependency identification approach has some desirable characteristics: (i) Undirected goal-dependencies can be found automatically between all goals in the goal tree, as the goals' formal specifications contain all the necessary information. (ii) The second step, (I2) may indicate the need for rewriting goal specifications in order to make them more precise. In the example above, although it was not possible to determine dependency direction due to few temporal operators in the specifications, the direction was established from domain knowledge. It may be beneficial in such cases to strengthen the formal specifications by introducing domain knowledge assumptions. In the example, the specification of RequestedFileFound could be modified by replacing Found(fl,rID) with \circ Found(fl,rID), and writing @Found(fl,rID) instead of Found(fl,rID) in RandomPeersForDataRoutingFound. More precise specifications derived from acceptable/verifiable domain assumptions arguably lead to higher quality requirements, further facilitating the identification of potential inconsistencies in the form of additional obstacles and conflicts.

The goal-dependency identification process often leads to the possibility of specifying a large number of goal-dependencies. To make the goal-dependency set readable, we remove those that are deducible by transitivity, giving its Hesse diagram.

Completeness of the Goal-Dependency Set. The condition for dependency existence ($\neg g_2 \Rightarrow \neg g_1 \ \mathbf{W} \ g_2$) can be used to verify the completeness of the goal-dependency set provided that system histories can be generated using e.g., model checking techniques. An informal, but practical, completeness criterion is that each goal in the goal tree is involved in at least one goal-dependency. Any goal that does not fulfill this condition is outside of the overall process that is to be realized by the MAS.

Linking Goal-Dependencies to MAS Nonfunctional Requirements. The aim of relating individual goal-dependencies to MAS nonfunctional requirements is to know the *type of vulnerability* that is generated by each goal-dependency. Similarly to the notion of vulnerability suggested in the context of agent-dependencies [31], goal-dependencies generate potential vulnerability: When a goal being depended upon is not achieved, the goal that depends on it will not be achieved. Consequently, failure of a goal may lead to the failure of the future MAS to operate according to the desired quality level.

The Step 1 of the proposed process involved, by application of the approach for reasoning about partial goal satisfaction [19], the identification of quality variables and their associated objective functions (to indicate whether the value of the variables should be maximized or minimized). For example, the quality variable `NumberOfRoutingPeers` is relevant for the `RandomPeersForDataRoutingFound` goal. Following [19] the specification given below can be written. That specification enriches the original goal specification with information about two quality variables that measure the degree of goal achievement. A quality variable can be conceptualized as kind of metric for measuring the degree to which a goal is achieved, whereas the sample space gives information on the case sample used to calculate probability values. The `NumberOfRoutingPeers` variable indicates that the probability of having at least two routing peers needs to be maximized, with a target value of 80%, while it is currently estimated at 30%. The second variable measures the probability of receiving a response from each peer regarding its availability for routing within a certain time frame.

Goal: `RandomPeersForDataRoutingFound`

Definition: The n random peers for data routing are found when n routing peers are found and each peer confirms availability.

ObjectiveFunctions:

Name	Def	Modal	Target	Current
<code>HighNumRoutingPeers</code>	$P(\text{NoRoutPeers} > 2)$	Max	80%	30%
<code>LowAvailRespTime</code>	$P(\text{AvailRespT} < 1\text{s})$	Min	70%	40%

QualityVariables:

`NumberOfRoutingPeers`: *Natural*

{ **Sample space**: set of routing peer numbers;

Def: number of peers that are used to route data between the sender and the receiver peers}

`AvailabilityResponseTime`: *Time*

{ **Sample space**: set of routing availability responses;

Def: time from the request for availability confirmation until the reception of the response}

FormalDef: $\forall p_s, p_r: \text{Peer}; fl: \text{File}; rID: \text{RequestID}; \text{Found}(fl, rID) \wedge \text{Sender}(p_s, fl, rID) \wedge \text{Receiver}(p_r, fl, rID) \wedge$
 $@(\exists p_1, \dots, p_n: \text{Peer}; \{p_s, p_r\} \cap \{p_1, \dots, p_n\} = \emptyset \wedge \diamond_{\leq 5s} \text{Available}(p_1, rID) \wedge \dots \wedge \diamond_{\leq 5s} \text{Available}(p_n, rID))$
 $\Rightarrow \diamond \text{RoutingPeers} = \{p_1, \dots, p_n\}$

In terms of system *nonfunctional requirements* (i.e., security, privacy, safety, usability, reliability, etc. – see e.g., [14], [15], [21]), the achievement of the above goal can be said to affect privacy and performance. In case the number of peers is small, it becomes easier to trace the entire route between the sender and the receiver, consequently allowing malicious users to obtain access to private peer information. Performance is affected in that, the more peers are used to route data, the less it is likely that the performance in terms of availability response time will be low. Consequently, the degree to which the goal is achieved affects the degree to which the above qualities are satisfied. As goal-dependencies are identified from properties common to goal pairs, each goal-dependency can be associated with a MAS quality, provided that the members of the dependency's `CommonProperties` attribute can be related to a quality

variable. In the example, if the `RandomPeersForDataRoutingFound` goal is involved in a directed or undirected goal-dependency that arises from the common property `RoutingPeers`, it can be inferred that this goal-dependency can be related to the degree to which privacy is satisfied in the MAS. In case this same goal is involved in a goal-dependency arising from the common property `Available(p,rID)`, the system performance is the quality to which this goal-dependency is related. The number of qualities to which a goal-dependency can be related is not restricted.

Relating goal-dependencies to nonfunctional requirements using common properties and quality variables allows taxonomy of system vulnerabilities to be proposed for the engineered MAS. If a goal-dependency is related to performance, this goal-dependency is said to generate *performance vulnerability*. Rich vulnerability taxonomies can be built from existing work in nonfunctional requirements analysis, such as [1], or standards (e.g., [14], [15]). To make explicit the specific vulnerability generated by a goal-dependency, the attribute `Vulnerability` is added to the goal-dependency specification template, as in the example below.

Goal-Dependency: `FindFileThenSearchForRoutingPeers`
Definition: Search for routing peers after the file to be transferred is found.
Involves: `RequestedFileFound`, `RandomPeersForDataRoutingFound`.
Direction: `RandomPeersForDataRoutingFound` **DependsOn** `RequestedFileFound`.
CommonProperties: `Found(fl,rID)`
Vulnerability: Reliability

Before the vulnerabilities can be used to generate and select between alternative goal to role allocations, the vulnerabilities need to be identified. This can be realized using the following process. For each goal:

- (VI1) Identify properties in the goal's formal specification whose values affect that goal's quality variables.
- (VI2) For each property identified in (VI1), check if there are goal-dependencies to which this property gives rise, and which involve the goal.
- (VI3) For each quality variable in the goal, identify system quality whose degree of satisfaction is measured by that quality variable.

For each goal-dependency involving the goal, indicate vulnerabilities by combining properties found in (VI2) with qualities found in (VI3) to which each property can be related.

2.2 Generate and Select between Alternative Goal-to-Role Allocations

The Step 3 generates and explores alternative allocations of goals to roles. Each allocation is a set of agent roles, such that each role is allocated a set of goals. The achievement of allocated goals becomes the responsibility of the agent that is selected to occupy the role. Choosing agents to occupy the defined roles is not treated.

Allocating Goals to Roles instead of Agents. Allocating responsibility for goal achievement directly to agents does not allow encapsulation and modularity nonfunctional requirements to be addressed satisfactorily when specifying requirements for large MAS. As suggested in [32], as soon as the complexity of MAS increases, modularity and encapsulation principles require MAS to be composed of agent roles. An agent can therefore play one or more roles to achieve goals within multiple and different agent organizations. In order to benefit from the *organizational* metaphor [32],

it is necessary to ensure the separation of agents' action execution characteristics from its expected behavior within MAS organizations.

In the Gaia MAS development framework ([32], [33]), a role is modeled as a set of responsibilities and permissions. Responsibilities are represented as protocols (i.e. activities that require interaction with other agents) that the role needs to execute, while role's permissions specify resources that the role can access and under which conditions. At a more abstract level, Gaia responsibilities can be seen as resulting from MAS goal operationalizations, involving, among other, the identification of agent capabilities and actions that are required for the goal to be achieved. Based on the role concept in Gaia, a restrictive way of conceptualizing a role is to consider it as being a set of MAS goals. While this is one of the many facets of the role concept used in MAS engineering, it may be sufficient to restrict the analysis during the RE step at this aspect of role only. Responsibilities and permissions could be derived from goals specifications later in the MAS development process. If the proposed goal-to-role allocation approach is to be used as the first requirements step in, e.g., Gaia, there are no barriers in enriching the suggested conceptualization with additional facets relevant for methodology-specific analyses.

The role conceptualization is consequently not fixed in the proposed allocation approach. It is up to the requirements engineer to choose the degree of expressivity of role by including its various facets (e.g., goals, permissions, etc.). The goal-to-role allocation approach does necessitate that the role be characterized *at least* as being a set of goals. Otherwise, alternative allocations cannot be studied.

Generate Alternative Roles. An alternative goal-to-role allocation is a set of roles such that all goals in the goal tree are allocated to at least one role. Using information about vulnerabilities, it can be shown that each allocation satisfies to a different degree the MAS qualities. Thus, the ultimate purpose of generating alternative allocations is to choose one that is considered as the most adequate by the RE stakeholders.

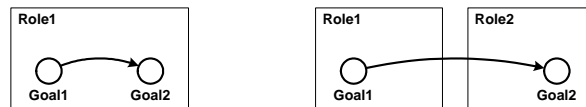


Fig. 1. Internalized goal-dependency (left) and externalized goal-dependency (right)

Internalization vs. Externalization. In economics and organization sciences, most of the analysis of distributing work between economic agents (such as, e.g., entire firms) has focused on the question of whether to realize activities internally, or to assign their responsibility to external agents (e.g., [5], [30], [24]).

In terms specific to the allocation approach proposed in this paper, the internalization decision results in a goal-dependency that is under the responsibility of a single role, i.e., the role contains both goals involved in the dependency (Fig. 1). Whether a goal-dependency is internalized or externalized will result in a different degree of MAS nonfunctional requirements satisfaction. For example, the externalization of a dependency may require interaction between distinct agents. This in turn could lead to worse response rates of the MAS, security issues resulting from the possibility of

interception of sensitive data communicated between the agents, etc. Consequently, it is assumed in the context of this paper, that a key parameter to consider when designing roles in MAS is whether to *externalize or internalize goal-dependencies within roles*. This is particularly relevant in the face of the long tradition economics and organization science preoccupation with internalization and externalization decisions, and when the organizational metaphor is adopted during MAS development.

Conclusions from seminal works in economics and organization science can be a valuable source of inspiration for justifying goal to role allocation decisions. The following motives can be used to argue for/against internalization decisions in the context of MAS development. Motives to internalize a goal-dependency (marked with “I”) can be:

- I-a. According to [5], it is the aim of exploiting *economies of speed* that pushes firms towards internalizing activities. Because firm throughput depends on uninterrupted flows of material and payments, precise planning and control is of paramount importance. As internalization implies that larger parts of the firm’s environment are under the influence of its management, it could be the strategy of choice for exploiting economies of speed [25]. A parallel can be made with the need for speed in MAS operation (or, more generally performance): To favor performance optimization in MAS, it is beneficial to internalize goal-dependencies, as the agent occupying the role will need to have capabilities allowing a larger part of MAS to be under its control.
- I-b. Internalization can reduce *transaction costs* (e.g., [30]), including the costs of finding, selling, negotiating, contracting, monitoring, and resolving disputes with other firms. Although it may be argued that transactions between MAS agents have no cost, this may not be the case if cost includes the impact of goal-dependency failure on the degree of MAS qualities satisfaction. Transaction cost between agents may be considered a function of vulnerabilities generated by the goal-dependency that is externalized and involves transaction between agent roles. To avoid the “cost” of the vulnerabilities, internalization may be the tactic of choice.
- I-c. In relation to the motives (I-a) and (I-b), if a transaction between firms involves *repeated interaction*, it may be better to internalize that transaction [30]. If two goals in a goal-dependency are likely to be achieved frequently during MAS operation, it may be beneficial to internalize that dependency. This allows, e.g., a role to be defined so that it can be occupied only by agents specialized in achieving the two goals, resulting in reduced vulnerability for qualities to which the internalized goal-dependency is related.

Motives to externalize a goal-dependency (marked with “E”):

- E-a. Internalization carries a *commitment to a particular way of doing business* [2], leading to lower flexibility of the firm in the face of changing environment conditions. Building MAS using complex roles requires very specific agents to be available to occupy the roles. In open MAS, where existing and new agents may enter and exit, increased internalization of goal-dependencies commits MAS to a particular way of functioning that may rapidly become obsolete.
- E-b. *Growth in the extent of the markets* reduces the incentives to internalize comparatively to externalizing activities through markets [17]. When designing MAS roles, it is relevant to consider externalizing goal-dependencies if this can allow the definition of roles that can be occupied by a wide variety of generic agents.

This may facilitate changing role occupancy during MAS operation, allowing, e.g., to replace dysfunctional agents by other available agents.

E-c. *Standardization of transactions* (contracts in particular) reduces the uncertainty a firm faces when externalizing activities (e.g., [17], [25]). In other words, the more predictable the transaction, the more likely it is to be externalized. In MAS, some goal-dependencies, if externalized may involve interactions that are standardized, that is, widely used patterns may exist to codify interactions. Vulnerabilities that appear in such goal-dependencies may be considered as having a limited impact.

The above considerations need to be perceived as starting points for discussion when choosing a goal-to-role allocation. They serve to justify decisions when generating alternative allocations.

A Process to Generate and Select Roles. The process described below can be used to generate alternative sets of potential MAS agent roles. The process starts by generating an initial allocation. Then, alternatives are created by changing the initial role set. For each goal-dependency in the goal tree:

- (GAR1) Identify the vulnerabilities generated by that goal-dependency.
- (GAR2) Discuss whether internalization or externalization would lead to increasing or reducing the probability of the vulnerability to occur. Base justifications on motives for internalizing or externalizing discussed above.
- (GAR3) Decide whether to internalize or externalize the goal-dependency.

Although the application of the process above results in a single set of roles, information about potential alternatives can be recorded during the application of the process. Consider Table 1 as an example of how decisions can be recorded during the application of the identification process.

Table 1. An example of how to record alternative role definitions

	Quality ₁	Quality ₂	Quality ₃	Quality ₄
Goal-dependency ₁	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> ++ -- </div> <div style="display: flex; justify-content: space-between;"> + + </div>	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> -- -- </div> <div style="display: flex; justify-content: space-between;"> + + </div>	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> - - </div> <div style="display: flex; justify-content: space-between;"> ++ + </div>	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> -- -- </div> <div style="display: flex; justify-content: space-between;"> + + </div>
Goal-dependency ₂	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> -- -- </div> <div style="display: flex; justify-content: space-between;"> ++ + </div>	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> - - </div> <div style="display: flex; justify-content: space-between;"> + + </div>	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> ++ ++ </div> <div style="display: flex; justify-content: space-between;"> + + </div>	<div style="display: flex; justify-content: space-between; border-bottom: 1px dashed black;"> -- -- </div> <div style="display: flex; justify-content: space-between;"> + + </div>

Each cell in the table is at the intersection of a goal-dependency and a MAS quality, and is separated in an upper and lower part. The upper part of a cell represents the impact of the decision to *internalize* a goal-dependency on the degree to which the concerned quality is satisfied in the MAS. The lower part of a cell represents the same information for the decision to *externalize* the goal-dependency. Each part of a cell may contain one of four symbols: (++) to indicate that the decision supports strongly and favorably the satisfaction of the concerned quality, (+) to indicate somewhat favorable support, (-) to indicate somewhat unfavorable support, and (--) to indicate strong unfavorable support. If the goal-dependency is unrelated to the quality (i.e., it does not generate a vulnerability for that quality), the cell is left blank.

Alternative roles can be constructed by choosing, in each non-blank cell, one cell part to indicate that the concerned goal-dependency needs to be internalized or externalized. As the table contains all possible individual alternatives (i.e., all possible internalizations and externalizations of goal-dependencies), it contains sufficient information to construct any alternative goal-to-role allocation.

While qualitative reasoning techniques, such as the one used to construct and interpret Table 1 have their limitations (notably in terms of accuracy [19]), they are accessible and are an adequate choice when too little information is available to provide *quantitative* motives (as opposed to *qualitative* ones proposed above).

3 Conclusions and Future Work

A systematic approach for allocating goals to agent roles during the RE step of the MAS development process is proposed. A novel type of inter-goal link, the *goal-dependency*, a type of the dependency relationship, serves two purposes in the approach. First, it is used to reason about the sequence of goal achievement in MAS, adding valuable information to classical goal refinement and contribution links. Second, each goal-dependency can be related, through goal quality variables and the value of the goal-dependency's *CommonProperties* attribute, to information about non-functional requirements of the MAS, to allow MAS vulnerabilities to be identified, classified for analysis, and used for agent role definition.

Two additional parameters for organizational design discussed in organizational sciences are the allocation of decision rights and the grouping of work in subunits of an organization. Further work is needed to study the tools and methods for integrating these factors in the process of designing MAS organizations. The proposed qualitative reasoning technique can be extended to integrate quantitative data. The use of goal-dependencies in the analysis of the timed operation of MAS during the RE step will also be addressed.

References

1. Anton, A., Earp, J., A. Reese, A.: Analyzing Website Privacy Requirements Using a Privacy Goal Taxonomy. In Proc. IEEE Int. Req. Eng. Conf. RE'02, 2002, 23-31.
2. Buzzel, R. D.: Is vertical integration profitable? Harvard Business Rev. (Jan.-Feb. 1983).
3. Caire, G., Coulier, W., Garijo, F., Gomez, J., Pavon, J., Leal, F., Chaainho, P., Kearney, P., Stark, J., Evans, R., Massonet, P.: Agent-Oriented analysis using message/uml. Proc. 2nd Int. Worksh. Agent-Oriented Softw. Eng. LNCS 2222, 2002.
4. Castro, J., Kolp, M., Mylopoulos, J.: Towards requirements-driven information systems engineering: the Tropos project. Inf. Systems, 27, 6 (2002) 365-389.
5. Chandler, A.: The Visible Hand - The Managerial Revolution in American Business. Cambridge, MA: Belknap Press, 1977.
6. Chung, L., Nixon, B., Yu, E., Mylopoulos, J.: Non-Functional Requirements in Software Engineering. Kluwer Publishing, 2000.
7. Cleland-Huang, J., Settimi, R., BenKhadra, O., Christina, S. Goal-Centric Traceability for Managing Non-Functional Requirements. Proc. Int. Conf. Softw. Eng., 2005.
8. Dardenne, A., van Lamsweerde, A., Fickas S.: Goal-directed requirements acquisition. Sc. Comp. Prog., 20 (1993) 3-50.
9. Devanbu, P.T., Stubblebine, S.: Software Engineering for Security: a Roadmap. In Proc. 22nd Int. Conf. on Softw. Eng., 2000.

10. Donzelli, P. A goal-driven and agent-based requirements engineering framework. *Req. Eng.*, 9 (2004) 16-39.
11. French, X.: Systematic Formulation of Non-Functional Characteristics of Software. In *Proc. Int. Conf. on Req. Eng. RE'98*, 1998.
12. Friedman, A., Camp, L. J.: Peer-to-Peer Security. In Bidgoli, H. (Ed.) *The Handbook of Information Security*. J.Wiley&Sons, 2005.
13. Fuxman, A., Liu, L., Mylopoulos, J., Pistore, M., Roveri, M., Traverso, P.: Specifying and Analyzing Early Requirements in Tropos. *Req. Eng.* 9, 2 (2004) 132-150.
14. IEEE Computer Society: IEEE Standard for a Software Quality Metrics Methodology. IEEE Std. 1061-1992, New York, 1992.
15. ISO: ISO/IEC Standards 9126-Information Technology-Softw. Product Evaluation. ISO, 1991.
16. Issarny, V., Bidan, C., Saridakis, T.: Achieving middleware customization in a configuration-based development environment: experience with the Aster prototype. *Proc. 4th Int. Conf. Config. Distr. Syst.*, 1998.
17. Langlois, R. N.: The vanishing hand: the changing dynamics of industrial capitalism. *Ind. and Corp. Change* 12, 2 (2002) 351-385.
18. Letier, E.: Reasoning about Agents in Goal-Oriented Requirements Engineering. Ph.D. Thesis, Univ. of Louvain, 2001.
19. Letier, E., van Lamsweerde, A.: Reasoning about Partial Goal Satisfaction for Requirements and Design Engineering. *Proc. SIGSOFT'04/FSE-12*, 2004.
20. Liu, L., and Yu, E. Designing information systems in social context: a goal and scenario modeling approach. *Info. Syst.*, 29 (2004).
21. Mylopoulos, J., Chung, L., Nixon, B. Representing and Using Nonfunctional Requirements: A Process-Oriented Approach. *IEEE Trans. on Soft. Eng.*, 18, 6 (1992) 483-497.
22. Al-Naeem, T., Gorton, I., Ali Babar, M., Rabhi, F., Benatallah, B.: A Quality-Driven Systematic Approach for Architecting Distributed Software Applications. *Proc. ICSE*, 2005.
23. Rolland, C., Souveyet, C., and Ben Achour, C. Guiding Goal Modelling Using Scenarios. *IEEE Trans. Softw. Eng.* (Dec. 1998).
24. Rubin, P. *Managing Business Transactions*. NY: Free Press, 1990.
25. Sturgeon, T. J.: Modular production networks: a new American model of industrial organization. *Ind. Corp. Change* 11, 3 (2002).
26. van Lamsweerde, A.: Goal-Oriented Requirements Engineering: A Guided Tour. In *Proc. 5th IEEE Int. Symp. Req. Eng.* 2001, 249-263.
27. van Lamsweerde, A., Letier, E.: Handling Obstacles in Goal-Oriented Requirements Engineering. In *IEEE Trans. on Soft. Eng.*, 26, 10 (2000) 978-1005.
28. van Lamsweerde, A. Goal-Oriented Requirements Engineering: A Roundtrip from Research to Practice. *Proc. 8th IEEE Int. Symp. on Req. Eng. RE, IEEE*, 2004.
29. DeLoach, S.A., Wood, M., Sparkman, C.: Multiagent system engineering. *Int. J. Softw. Eng. Knowl. Eng.* 11, 3 (2001) 231-258.
30. Williamson, O. *The Economic Institutions of Capitalism*. NY: Free Press, 1985.
31. Yu, E. Modeling Strategic Relationships for Process Reengineering. Ph.D. Th., Univ. of Toronto, 1995.
32. Wooldridge, M., Jennings, N.R., Kinny, D.: The Gaia methodology for agent-oriented analysis and design. *J. Auton. Ag. M.-Ag. Syst.*, 3, 3 (2000) 285-312.
33. Zambonelli, F., Jennings, N.R., and Wooldridge, M. Developing Multiagent Systems: The Gaia Methodology. *ACM Trans. on Softw. Eng. and Meth.*, 12, 3 (2003) 317-370.