Reasoning About Willingness in Networks of Agents

S. Dehousse[1], S. Faulkner[1], H. Mouratidis[2], M. Kolp[3], P. Giorgini[4]

[1]Information Management Research Unit, University of Namur, Belgium
{stephane.dehousse, stephane.faulkner}@fundp.ac.be
[2] Innovative Informatics, School of Computing, Univ. of East London, England
h.mouratidis@uel.ac.uk
[3] Information System Unit - University of Louvain, Belgium
kolp@isys.ucl.ac.be
Department of Information and Communication Technology, University of Trento
giorgini@dit.unitn.it

Abstract. The i* Strategic Dependency model has been successfully employed to analyze trust relationships of networks of agents during the early stages of multiagent systems development. However, the model only supports limited trust reasoning due to its limitation to deal with the vulnerability of the depender regarding the failure of the dependency. In this paper, we introduce the concept of willingness, which provides a solution to the above problem and therefore allows a more complete analysis and reasoning of trust relationships in networks of agents.

1 Introduction

An interesting challenge for the agent research community is the development of large-scale dependable Multiagent Systems. As stated in the Call for Papers (CFP) for the SELMAS'06 workshop, "the dependability of a computing system is its ability to deliver service that can be justifiably trusted". In multiagent systems, which consist of network of agents, this means that the developer needs to analyze explicitly the trust relationships between the different agents of the network/system.

The i* Strategic Dependency (SD) model has been employed to model trust relationships between agents and in many cases has been stated for its appropriateness to explore trust relationships during the early stages of a multiagent system's development. The rich modeling concepts supported by the model provide a better basis to explore the broader implications of trust relationships than conventional non-intentional models, such as data flow diagrams and/or object-oriented analysis languages (e.g. UML). Moreover, the model does not treat trust as an isolated concept with special semantics but in parallel with the rest of the system's goals. Additionally, the model facilitates the analysis of trust-related issues within the full operations and social context of the system-to-be and it also supports trade-off analysis of trust and other competing quality requirements such as performance.

The SD model [9] is used to construct a network of social dependencies amongst actors, where an actor represents an entity such as an agent that has intentionality and strategic goals within the information system or within its organisational setting. In particular, the model is a graph, where each node represents an actor, and each link between two actors indicates that one actor depends on another for something in order that the former may attain some goal. A *dependency* describes an "agreement" (called *dependum*) between two actors: the *depender* and the *dependee*. The depender is the depending actor, and the dependee is the actor who is depended upon. The SD supports different types of dependencies describing the nature of the agreement. *Goal dependencies* are used to represent the transfer of responsibility for fulfilling a goal. *Softgoal* dependencies are similar to goal dependencies, but their fulfilments cannot be precisely defined (for instance, the appreciation is subjective or the fulfilments can occur only in a given extent); *task dependencies* are used in situations where the dependee is required to perform a given action; and *resource dependencies* require the dependee to provide a resource for the depender.

However, the SD model demonstrates a limitation related to the vulnerability of the depender regarding the failure of the dependency[1]. Such limitation restricts developers in performing a full reasoning about the trust relationships of the system-to-be. In this paper we present an approach based on the concept of willingness to overcome this limitation.

It is worth mentioning that due to lack of space, we have decided to adopt a simplified notation for the purpose of this paper. In particular, agents are denoted by the set *Ag* noted as a, b,… $\in$ *Ag*. We note the set of services *S* and a particular service as s_x where $s_x \subseteq S$.

- *depender(a, s_x)* means that *a* is depender for the service s_x.
- *dependee(a, s_x)* means that a is dependee about the service s_x.
- *depends(a, b, s_x)* means that a is depender, b is dependee about the service s_x

The rest of the paper is organized as follows. Section 2 discusses the vulnerability limitation inherent to *i**. Section 3 introduces the concept of willingness and defines its different constituent elements, whereas Section 4 illustrates the concepts and limitations with the aid of scenarios. Section 5 discusses how willingness can be positively influenced to strengthen a dependency and Section 6 proposes the introduction of the delegation relationship. Finally, Section 7 presents related work while Section 8 concludes the paper and briefly presents future works.

2 The Down-side of a Dependency

Although the Strategic Dependency model has been successfully employed to model trust relationships of networks of agents during the early stages of multiagent systems development, it supports limited trust reasoning due to its limitation to deal with the vulnerability of the depender regarding the failure of the dependency [9] (we call this the down-side of a dependency). This is an important issue since potential failure of dependencies may not only hurt the depender, but it may set off a perilous

[1] We define failure of a dependency as the situation in which a dependee fails or refuses to satisfy the dependency.

chain reaction that would endanger the whole system. This prevents developers to fully reason about the trust between the agents of the system, which results in incomplete analysis of the trust relationships of the network.

In general, the above mentioned limitation is influenced by two elements. The first element is the vulnerability of the depender which is an intrinsic property of the dependency for the depender. The second element, that we call "failure of the dependency" is directly related to the dependee(s).

The next sections discuss these two components of the "down-side" of a dependency and present some definitions of the related concepts.

2.1 The Depender's side

In our work, and in order to sustain our analysis of the first constituent of the "down-side" of a dependency, we define the concept of vulnerability as follows:

Vulnerability is a characteristic of the depender that causes the agent to suffer of some incapability to achieve its goals as a result of the dependee's failure to achieve the dependum.

This definition emphasizes the concept of vulnerability as an internal quality of the depender in a dependency. In other words, the vulnerability of an agent, with respect to the dependum, is only related to the importance of the dependum for that agent.

The i* SD model [9, 10] distinguishes three degrees of strength (importance) for a dependency that can be applied independently on each side: Open ("O"), Committed (unmarked) and Critical ("X"). The degree of strength for the depender corresponds to its level of vulnerability about the dependum. In an *Open Dependency*, failure of the dependum would have no serious consequences on the depender and therefore the depender's vulnerability is low. In a *Committed Dependency*, failure of the dependum might significantly affect the depender's goals and therefore the depender's vulnerability is average. In a *Critical Dependency* failure of the dependum would seriously affect the depender's goals and therefore the depender's vulnerability is high.

As a response to the "down-side" of a dependency, Yu [9] suggests three mechanisms: enforcement, assurance and insurance. An Enforcement mechanism consists of finding some way for the depender to cause some goal of the dependee to fail, e.g. if there is a reciprocal dependency. Assurance refers to a situation where there is some evidence that the dependee has an interest in delivering the dependum, e.g. if the fulfilling of the commitment is in the dependee's own interest. It is worth mentioning that these two measures would only have an impact on the dependee's behaviour about the dependum and internally the depender's vulnerability would never be mitigated. Finally, insurance mechanisms are supposed to reduce vulnerability of a depender by reducing the degree of dependence on a particular dependee. The consequence of such measure is not a mitigation of the depender's vulnerability, but rather a potential increasing of the probability that the dependum will be achieved. Conversely to Yu's claim, all these mechanisms only contribute to fortifying a dependency, and do not help to mitigate vulnerability. Effective measures to mitigate vulnerability should rather try to influence importance of the dependum for the depender, e.g. by creating alternatives to the dependum or its parent goal(s) internally at the depender's side.

2.2 The Dependee's side

This second element catches the dependee's influence on the "down-side" of the dependency. In particular, the factors, at the dependee's side, which contribute to the success or the failure of the dependency are analysed. To enable the success of a dependency, the dependee must have at least three qualities regarding the dependum: ability, authorization and willingness. The ability and authorization qualities of a dependee have been previously discussed by Giorgini et al. [6]. However, the dependee's willingness about the dependum remains an open question. The next section presents a detailed discussion of the dependee's willingness and its different constituents.

3 Dependee's Willingness

The *willingness (W°)* of an agent about a dependum expresses its intrinsic readiness to actually fulfil the dependum. It is based on the combination of three elements: the criticality (C°) of the dependum for the dependee; the pressure (P°) on the dependee about the dependum; and the reciprocity (R°) with the depender(s). The willingness of an agent involved in the system can be derived for a specific goal, task or resource. The impact of the different constituent elements is weighted by weight parameters (α,β,γ) according to the domain application. These parameters enable the designer to adjust influence of the different factors to better suit the context of the implementation, i.e. a β or γ greater than α corresponds to reputation-based systems or systems with a high degree of cooperation. Moreover, to be able to compare values computed for different agents, the willingness is constrained between 0 and 1 by imposing the different factor's values range from 0 to 1 and the sum of the weight parameters equal to 1.

$$W°(a, s_x) = \alpha * C°(a, s_x) + \beta * P°(a, s_x) + \gamma * R°(a, s_x) \qquad \text{where } \mathit{dependee}(a, s_x)$$

A running example is used in the rest of the paper, to assist readers in understanding these elements. The example forms a view of a substantial case study. In this example, four agents are identified: **Bob** is purchasing manager, **Alice** and **Jos** are accountants and **Bert** is stock manager. Additionally, for sake of simplicity, in this approach, the notion of service is used to refer to a goal, a task, or a resource.

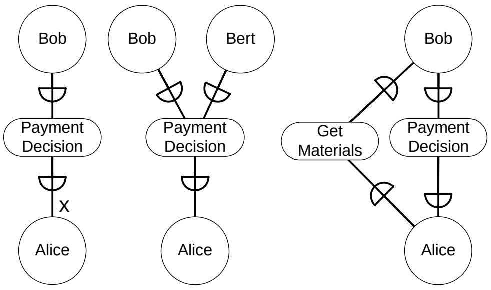


Fig. 1. a. Criticality - b. Pressure - c. Reciprocity

3.1 Criticality

The criticality (C°) factor indicates the degree of importance a service has for an agent. This importance is based on the value of the service for the agent intrinsically, i.e. apart from any claim of other agent(s). The achievement of a dependum may be critical for a dependee, for different reasons. For instance when the dependee has some goals related to the achievement of the dependum.

Example 1 *According to the company procedures, Alice is responsible for the accounting of the managers. She needs achievement of "payment decision" for each manager to do the accounting. Bob must have the payment decision about its order given by an accountant of the company. So Bob seek for Alice payment decision (Fig.1.a).*

In example 1, the goal of Alice, "do the accounting" is linked to the achievement of "payment decision" service. As a consequence, this service turns to be critical for Alice. These circumstances increase her willingness about its achievement. Through criticality analysis we have quantified some evidence that the dependee, Alice, has some interest to fulfill, apart from any claim of the depender, Bob.

Decision about the level of criticality of a service for an agent is taken by the designer.

C°(a, sx) in [0,1] where dependee(a, sx)

3.2 Pressure

The *pressure (P°)* factor indicates the degree of influence that a group of dependers (targeting the same dependum) has on the dependee's behaviour. It is an external factor that impacts the dependee's willingness about the achievement of the dependum.

Example 2 *According to the company procedures, Bob needs a payment decision on its order and Bert needs payment decision to decide on the entry of an item in the stock. Yet, this decision can only be given by a company's accountant. So Bob and*

Bert seek for Alice payment decision. (Fig.1b)

The dependency in example 2 has two dependers (Bob and Bert) about a dependum "payment decision" depending on Alice. Alice is therefore under pressure of Bob and Bert about this service. This pressure increases her willingness to fulfil the dependum.

To refine our analysis, different levels of pressure can be identified according to the relative position occupied by a depender. For example, in real working conditions, the pressure imposed by Bob, the purchasing manager, can be considered greater than the pressure imposed by Bert, the stock manager. Consequently, the global pressure becomes the sum of the weighted individual pressure of the dependers involved. The weight (*p*) given to a position is determined by the designer or according to the application domain.

$$P°(a,s_x) = 1 - 1/\exp(\sum p_i * ag_i)$$

where dependee(a, s_x) and

$$ag_i \in Ag \,|\, depends(ag_i, a, s_x)$$

3.3 Reciprocity

The *reciprocity (R°)* factor indicates the influence of relations of mutual dependence between the dependee and some depender(s).

Such reciprocal relationship makes the dependee, at her turn, vulnerable to the behaviour of the depender. Considering that agents basically follow rules of tit-for-tat, a situation of reciprocal relationship should positively influence the behaviour of the dependee agent about the fulfilment of the dependum.

Example 3 *According to the company procedures, the purchase manager is responsible for office materials order for all employees of the company. Therefore, Alice needs Bob to get her office material whereas Bob needs accountant payment decision on its order. So Bob seeks for Alice's payment decision. (Fig.1c)*

In example 3, there is a relation of mutual dependence between Bob and Alice. As Alice is depending upon Bob, she would rather adopt behaviour in favour of Bob to positively influence Bob's behaviour concerning her request. As agents adopt a tit-for-tat strategy, a reciprocal relationship increases the willingness of the dependee about the fulfilment of the dependum.

Moreover, we may reasonably argue that the more critical the dependum of the reciprocal dependency is for the dependee, the more this reciprocity increases its willingness. Therefore, the reciprocity factor is not only based on the number of mutual dependencies but also on their respective criticality for the dependee. In the example, if the criticality of "get material" service increases for Alice, her willingness about "payment decision" will be greater.

The formulae below can be used to determine the pressure that some depender(s) impose on a dependee.

$$R°(a,s_x) = 1 - \frac{1}{\exp\{\sum C°(a,s_y)\}}$$

where $s_x, s_y \in S, a, b \in Ag \,\middle|\, depends(b, a, s_x) \wedge$
$depends(a, b, s_y)$

The reciprocity factor is directly related to the depender's claim; it turns into figures the ability of the depender to cause some goal of the dependee to fail

4 Scenarios

When agents need to be involved in a dependency, they should trust the dependee. This trust reflects its estimation on the willingness of the dependee to personally fulfil the dependum. The previous section has presented the different elements that could help to determine this value. At the end of the estimation of a dependee's willingness about a service, the depender may decide that the willingness value is high enough for the dependency to remain unchanged or that the value is low. In the second case, the depender should try to improve willingness value. One solution consists in positively influencing, through specific measures, the determinants of this value: criticality, pressure or reciprocity. To sustain the presentation of such measures and to demonstrate the applicability of our work, we present three scenarios, based on the running example case study, which emphasize different dependency's settings with variations on dependees' side.

Scenario 1 is the simplest dependency's configuration scenario which involves one depender and one dependee about a unique dependum.

Scenario 1 *According to the company procedures, Bob must have the payment decision about its order given by an accountant of the company. So Bob seeks for Alice's payment decision (Fig.2a).*

Scenarios 2 and 3 illustrate situations of one depender with several dependees.

Scenario 2 *According to the company procedures, Bob needs a decision on its payment order. Yet, this decision can only be given by one of the company's accountants. So Bob seeks for Alice* **or** *Jos payment decision. (Fig.2b)*

Scenario 3 *According to the company procedures, Bob needs a decision on its payment order. Yet, this decision must be given by two company's accountants. So Bob seeks for payment decision of Alice* ***and*** *Jos. (Fig.2c)*

Although these scenarios might seem similar, they emphasise differences in the dependencies between the agents of the system. For instance, in scenario 2 Bob can obtain payment decision from either Alice or Jos. Conversely, in scenario 3, Bob must have the payment decision fulfilled by the intervention of both Alice and Jos.

The i* SD model does not provide concepts and notations to emphasize such distinctions. Therefore, we propose to lightly extend the SD model of i* with concepts that enables the modelling of such situations. In particular, we define the following:

A dependency with substitute dependees means that each dependee is able to fulfil alone the dependum.

A dependency with complementary dependees means that contribution of all dependees is required to fulfil the dependum.

The Bob, Alice and Jos relationship derived from scenario 2 corresponds to a dependency with substitute dependees. Indeed, Alice and Jos are both able to fulfil alone the payment decision service.

Conversely in scenario 3, neither Alice nor Jos can achieve alone fulfilment of payment decision service. As such, scenario 3 is a good illustration of a dependency with complementary dependees.

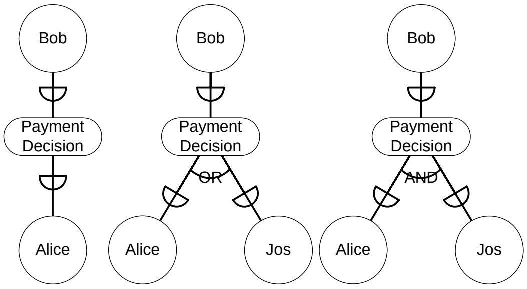


Fig. 2. a. One Dependee - b.Substitute Dependees - c. Complementary Dependees

5 Willingness Measures

5.1 One depender-One Dependee

In scenario 1, the willingness of Alice about "payment decision" dependum is clearly poor. The service is not critical at all for her, the pressure comes only from one depender and there is no reciprocal relationship. To improve her willingness, developers can firstly try to influence the criticality of the service for her.

According to the definition of the criticality, a solution consists in creating a relation (precondition) between a goal of Alice and the "payment decision" dependum. This can be achieved by the introduction of a new condition that relates one of Alice's goals to the dependum. For instance, to achieve her goal "do accouting", Alice must have "payment decision" fulfilled (Fig.1a).

If it is not possible to increase criticality or not enough; developers could try to increase pressure on the dependee. Following example 1, Alice has only one depender. Adding a new depender will contribute to increase her willingness. In his position of stock manager, now, Bert has to have "payment decision" achieved in order to authorize or not the entry of ordered item in the stock. This situation implies that Bert becomes an additional depender of the Bob-Alice dependency (Fig.1b).

Finally, if previous measures are not possible or do not produce the desired results, developers could employ the reciprocity factor. In this situation, a reciprocal dependency is added between Alice and Bob. For example, according to the company procedures, the purchase manager is responsible for ordering office materials for all employees of the company. As a consequence, Alice relies on Bob to obtain the office material she needs. This situation implies the creation of a reciprocal dependency between Alice and Bob about the dependum "get office material" (Fig.1c).

5.1 One depender-Substitute Dependees

In the second scenario, Bob seeks consent from Alice or Jos about the “payment decision” dependum. In the i* SD model, such situation leads to a dependency between Bob and Alice or Jos, where Alice and Jos are substitute dependees. As Bob may rely either on Alice or Jos for its dependum, we should evaluate both the willingness of Alice and Jos.

The analysis of the willingness of Alice and Jos is quite similar and lead to the conclusion of a poor willingness about the service. For both Alice and Jos, the service is not critical, the pressure comes only from one depender, and there is no reciprocal relationship.

As Alice and Jos are substitute dependees, the global willingness about the dependum for the depender Bob is the greatest one. Therefore to improve global willingness, we can chose to try to improve willingness of Alice, willingness of Jos or even both. To enable comparison with scenario 1, we focus on the solutions to increase Alice’s willingness.

First option is to increase for Alice the criticality of the service. As the criticality is based on the value of the service for the agent intrinsically, the presence of another dependee should have no impact on the measures that could be taken. We can therefore use the same measure as for scenario 1, i.e. introduce a new constraint that states to achieve her goal “do accouting”, Alice must have “payment decision” fulfilled (Fig.3a). Thanks to the new constraint, the “payment decision” service becomes critical for Alice. In example 4, conclusion of this measure was an increasing of the criticality factor of Alice about the dependum.

But, due to the introduction of a new dependee, this measure appears to have another consequence on the relationships between Bob, Alice and Jos. As Jos is also able to fulfil Alice’s critical service, Alice could initiate a dependency on Jos about it, in order for her to easily or better achieve her related goal, “do accounting”. We can consider that Alice is becoming an additional depender on the dependency Bob-Alice or Jos (Fig.3b). It increases the pressure factor of the other dependee(s), i.e. Jos, about the dependum. As a consequence, by making the dependum critical for a dependee, in a situation of substitute dependees, we not only have increased this dependee willingness but also the other dependee(s) willingness through an increasing of the pressure they face.

In a situation with substitute dependees another measure may be used to increase criticality of the dependum for a dependee. It consists in creating incentives for the dependee about personnaly achieving the dependum.

Scenario 4 *Alice receives a bonus for each payment decision. Alice wants to increase her personal payoff. So she has interest in achieving herself “payment decision”. Bob can seek for Alice or Jos about payment decision.*

In the above scenario, Alice has now great interests in being the one that actually achieve “payment decision” while the situation of Jos is unchanged. It results in a situation of partial competition between Alice and Jos, indeed incentives are only on Alice’s side.

Now, if we also create incentives for Jos to personally achieve “payment decision”, the competition becomes full. Configurations of competition between substitute dependees may considerably reduce chances of dependency failure for the depender.

If it is not possible to increase criticality or not enough, then we could try to increase pressure on the dependees. As for the criticality factor, we can reemploy the measure used in scenario 1, i.e. introduce an additional depender, Bert (Fig.3c). Such measure will always affect all dependees while its respective impact is based on the position criteria. Therefore, we have not only achieved increasing the pressure on Alice but also on Jos.

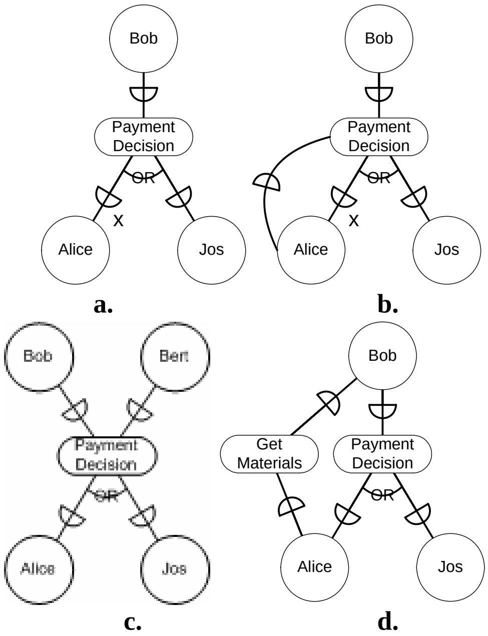


Fig. 3. Scenarios with Substitutes Dependees

Finally, if previous measures are not possible or enough, we could act on the reciprocity factor. Like in scenario 1, we create an internal situation that implies the creation of a reciprocal dependency Alice-Bob about the dependum “get office material” (Fig.3d). The reciprocity factor of Alice has increased while Jos’ is unchanged. A situation of substitute dependees does not affect measures related to the reciprocity factor.

To sum up, with the use of a number of different scenarios we have demonstrated that a situation of substitute dependees does not influence measures on pressure and reciprocity factor. Yet, it affects measures related to the criticality factor. In a dependency with substitute dependees, the global willingness about the dependum is the maximum of the willingness of the dependees.

5.1 One depender-Complementary Dependees

In scenario 3, Bob seeks the consent of Alice and Jos about “payment decision”. In the i* SD model, this situation leads to a dependency Bob-AliceandJos where Alice and Jos are complementary dependees. As Bob have to rely on Alice and Jos for its dependum, we should evaluate both the willingness of Alice and Jos.

The analysis of the willingness of Alice and Jos indicates that both of them demonstrate a poor willingness about the service. For both Alice and Jos, the service is not critical at all, the pressure comes only from one depender and there is no reciprocal relationship.

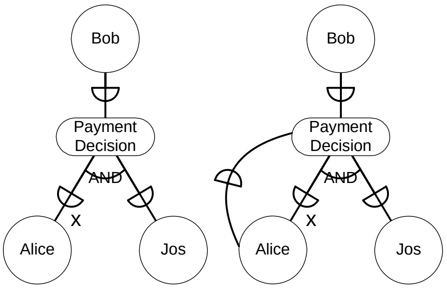
As Alice and Jos are complementary dependees, the global willingness about the dependum for the depender Bob is the smallest willingness among the dependees. Therefore to improve global willingness, we have to improve willingness of all dependees, starting by the weakest. To enable comparison with previous scenarios, we consider that Alice is the weakest and therefore starts with solutions to increase Alice's willingness.

To increase Alice's willingness, we first try to influence her criticality factor. Like in previous scenarios (1 and 2), we introduce a new situation to create a link between one of her goals and the dependum (Fig. 4.a and 4.b). Consequences of this measure are the same as for scenario 2, i.e. an increasing of Alice's criticality factor and an increasing of the pressure on other dependee(s), i.e. Jos. Yet contrary to scenario 2, by definition, no competition settings can happen among complementary dependees.

After criticality factor, we try to raise pressure on Alice about the dependum. As in scenario 2, increasing the pressure by the addition of a new depender impacts all dependees, i.e. Alice and Jos (Fig. 4.c).

Finally to increase the reciprocity factor of Alice, we create a new relationship between Alice and Bob (Fig. 4.d). This situation affects Alice's reciprocity factor without any impact on Jos' willingness factor.

In a dependency with complementary dependees, the global willingness about the dependum is the minimum of the willingness of the dependees. Consequently, measures to improve it should try to increase factors for dependee(s) with the lowest willingness.



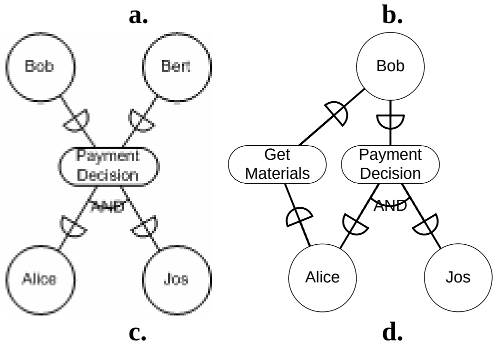


Fig. 4. Complementary Dependees

6 Delegation Measures

In the previous section, we have analyzed measures to improve willingness through its different constituent elements. If these measures are still not enough to ensure minimum trust of the depender in dependency's success, the depender may transform its dependency into a constraining delegation: delegation of obligation.

A delegation of obligation gives an imperative order from the delegator on the execution, the access to or the fulfillment of the delegatum [4]. The delegator corresponds to the depender of the dependency and the delegatum is an expression of the dependum.

In scenario 1, if all measures to improve willingness of Alice have failed, we can set up a delegation between Alice and Bob about the dependum. Concretely, Bob makes a positive delegation of obligation on Alice about the service "give payment decision". A positive obligation means that Alice is force to do something, opposite to negative obligation force to not do. Moreover, in scenario 1, Bob has only one dependee, no alternative exists. Turned into a delegation, this situation leads to a blind delegation as the delegator has not sufficient information on the unique delegatee to form a trust opinion. Compare to other forms of delegation, a blind delegation will require a monitor in order to compensate the lack of trust in the delegatee. Bob would therefore add a monitoring agent on its delegation to Alice.

In scenario 2 with substitute dependees, a delegation of obligation from Bob on Alice about the dependum will compensate lack of trust in dependency's success (Fig. 5).

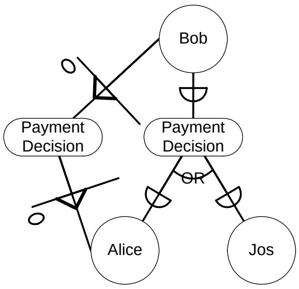


Fig. 5. Substitute dependees enforced by delegation of obligation

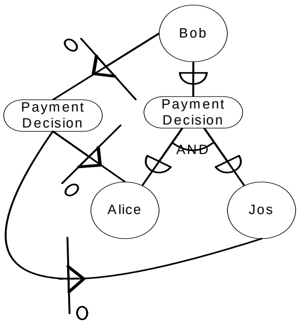


Fig. 6. Complementary dependees enforced by delegation of obligation

In scenario 3 with complementary dependees, the delegation of obligation must target all dependees, i.e. Alice and Jos, in order to compensate lack of trust in dependency's success. Indeed, imposing a delegation only on some dependees would be inefficient as the trust in dependency's success corresponds to the weakest willingness of the group of dependees (Figure 6).

7 Related Work

In the i* SD model, it is assumed throughout the analysis that the dependee will honour the dependency. However, this is not always the case meaning that the depender becomes vulnerable to the failure of the dependency [9]. As a consequence to the presence of such "down-side" of a dependency, evaluation of trust in the dependee about the dependum is crucial.

The line of work initiated by Castelfranchi and Falcone [3] has highlighted the importance of a cognitive view of trust (particularly for Belief-Desire-Intention agents [8]). To evaluate the trust an actor place in another actor about a service, different beliefs related to the motivations of the agent are considered: competence, willingness, persistence and motivation. For example, the competence belief refers to the agent capability to the deliver the service. Additionally, in recent work, Giorgini et al. [5, 6], borrow the notion of capability to address the problem of trust in dependency. Their approach states that if an agent has both permission and capability about a service, than the depender can be confident about dependency's success. Permission can be the result of ownership or delegation while capability could also be delegated. While this approach is quite interesting, it just gives a partial view on trust in dependency, because, as noticed by Castelfranchi and Falcone, capability is only a pre-requisite to trust. Our work complements this approach by dealing with the impact of the other beliefs (willingness, persistence and motivation) on dependency's failure. For sake of simplicity, we use the notion of agent's willingness that regroups through its different determinants the characteristics of these beliefs.

In the original work on i* SD model [9], the "down-side" of a dependency has been treated mainly through the concept of vulnerability. Surprisingly, this concept has not been clearly defined. More recent papers [7, 10] state that dependency relationships bring vulnerabilities to the system and the depending actor (the depender). However, our work has demonstrated that vulnerability of the depending actor is not a consequence of the dependency but rather an intrinsic property of the agent. Therefore, to clarify the situation, we have suggested a new definition for the concept of vulnerability. Moreover, we have suggested that the measures presented by Yu [9] to mitigate vulnerability were in fact measures to fortify the dependency by influencing dependee's behaviour.

8 Conclusion and Future Works

In this paper we have argued that in order to fully reason about trust during the development of multiagent systems, developers should consider the willingness of a dependee to fulfill the dependum. We have also described an approach to reason about willingness of agents based on the concepts of criticality, pressure and reciprocity. Our approach provides a first solution to the vulnerability limitation demonstrated by the i* SD model, and therefore allows developers to reason about trust in a structured way.

Our work is still at an exploratory stage. The proposed approach has been applied to various examples of case studies domains from the literature but it still remains to be applied in a large-scale real-life case study. It is worth commenting on the scope and the domain characteristics for which we believe the presented approach is appropriate. It is intended that the proposed process is performed by a software engineer (or software team), during design time, and not from software agents during run-time. We envisage the approach to be suitable for a large number of agent-based applications, where it is possible to identify stakeholders and their dependencies and where vulnerabilities of actor dependencies play an important role for the realization of the system's goals. As such, we believe that our approach is not suitable for the development of embedded software or system software (operating systems for instance) since in such systems there are no identifiable stakeholders. Moreover, due to lack of automated tool support, and the difficulty in considering manually all the possible conflicts identified during the vulnerability analysis, we believe that our approach is suitable for analyzing small to medium size agent based systems of up to 100 agents. We anticipate however that tool support will extend the applicability of the presented approach to large-scale real world agent-based applications.

This paper describes work in progress and future work includes the implementation of automatic tool support for the proposed approach as well as the development of a methodology to help computation of the willingness determinants based on refined formulae. We also plan to investigate solutions at the SD or SR levels that mitigate depender's vulnerability. In particular, we believe it would be interesting to consider the introduction of new goals or softgoals that could impact depender's vulnerability.

References

1. M. Blaze and J. Feigenbaum and A. D. Keromytis: The Role of Trust Management in Distributed Systems Security, In *Proc. of Secure Internet Programming* (1999) 185-210

2. J. Carter and E. Bitting and A. A. Ghorbani: Reputation Formalization within Information Sharing Multiagent Architectures, In *Proc. of Computational Intelligence* (2002) 45-64

3. C. Castelfranchi and R. Falcone: Principles of trust for MAS: cognitive anatomy, social importance, and quantification, In *Proc. of Int. Conf. of Multi-Agent Systems (ICMAS'98)* (1998) 72-79

4. S. faulkner and S. Dehousse: A Delegation Model for Designing Collaborative Multi-agent Systems, In *Proc. of 9th Int. Conf. on Knowledge-Based Intelligent Information and Engineering Systems (KES'05)*, R. Khosla, R. J. Howlett, L. C. Jain (Ed.), Lecture Notes in Computer Science, Vol. 3682, Springer-Verlag GmbH, Melbourne, Australia (September, 2005) 858.

5. P. Giorgini and F. Masscci and J. Mylopoulos and and N. Zannone: Filling the gap between Requirements Engineering and Public Key/Trust Management, In *Proc. of 2nd Int. Conf. on Trust Management (iTrust'04)* (2004).

6. P. Giorgini and F. Masscci and J. Mylopoulos and and N. Zannone: Modeling Security Requirements Through Ownership, Permission and Delegation, In *Proc. of 13th IEEE Int. Conf. on Requirements Engineering (RE'05)*, IEEE Computer Society Press, Los Alamitos, California (2005).

7. L. Liu and E. S. K. Yu and J. Mylopoulos: Security and Privacy Requirements Analysis within a Social Setting, In *Proc. of 11th IEEE Int. Conf. on Requirements Engineering (RE'03)* (2003) 151-161.

8. M. Wooldridge: *An Introduction to MultiAgent Systems*. John Wiley and Sons, Chichester, England, (2002).

9. E. S. K. Yu and J. Mylopoulos: From E-R to "A-R" - Modelling Strategic Actor Relationships for Business Process Reengineering, In *Proc. of 13th Int. Conf. on the Entity-Relationship (ER'94)*, Pericles Loucopoulos (Ed.), Lecture Notes in Computer Science, Springer (1994) 548-565.

10. E. S. K. Yu and L. Liu: Modelling Trust for System Design Using the i* Strategic Actors Framework, In *Proc. of Trust in Cyber-societies: Integrating the Human and Artificial Perspectives*, R. Falcone, M. Singh, Y.-H. Tan (Ed.), Lecture Notes in Computer Science, Vol. 2246, Springer-Verlag GmbH (2000) 175-194.