Computer Standards & Interfaces xxx (2010) xxx–xxx

Contents lists available at ScienceDirect

Computer Standards & Interfaces

journal homepage: www.elsevier.com/locate/csi

Trust establishment in the formation of Virtual Organizations

Anna C. Squicciarini [a,*], Federica Paci [b], Elisa Bertino [c]

[a] College of Information Sciences and Technology, The Pennsylvania State University Park, PA, USA
[b] Dipartimento di Ingegneria e Scienza dell'Informazione Università di Trento, Trento, Italy
[c] Cerias and Computer Science Department, Purdue University, West Lafayette, IN, USA

ARTICLE INFO

Available online xxxx

Keywords:
Trust negotiation
Virtual Organization
Semantic of trust

ABSTRACT

Virtual Organizations (VOs) represent a new collaboration paradigm in which the participating entities pool resources, services, and information to achieve a common goal. VOs represent an interesting approach for companies to achieve new and profitable business opportunities by being able to dynamically partner with others. Thus, choosing the appropriate VO partners is a crucial aspect. Ensuring trustworthiness of the members is also fundamental for making the best decisions. In this paper, we show how trust negotiation represents an effective means to select the best possible members during different stages in the VO lifecycle. We base our discussion on concrete application scenarios and illustrate the tools created by us that integrate trust negotiation with a VO Management toolkit.

1. Introduction

The evolution of Internet technology has promoted collaborations between different organizations. The Web 2.0 embracing new collaborative applications, indicates a new "social" approach to generating and distributing Web content, characterized by open communication, decentralization of authority, and freedom to share and re-use. The formation of dynamic coalitions aiming at sharing services can benefit from Semantic Web technologies to facilitate cooperation across entities and peers that use different platforms and systems. Also, VO members, by taking advantage of Semantic Web languages, can easily integrate and share data.

In such context, Virtual Organizations (VOs) represent an important collaboration paradigm allowing the participating entities (enterprises or individuals) to pool resources, services, and information to achieve common business goals. VOs are often created on demand and dynamically evolve over time. A VO is typically initiated by one or more organizations, also in charge of establishing collaboration policies through formally specified collaboration contracts. Additional members are then added to the initial VO, depending on the VO's goal and evolution. Because VO's members may have to share, combine and integrate sensitive information, trustworthiness of the members is fundamental. Especially in the context of Semantic Web applications, data integration must occur from data provided by trusted entities, which trustworthiness can be verified upon VO formation.

Forming a trusted VO presents significant challenges, due to the heterogeneous nature of the VO participants, the lack of standardized trust establishment approaches and the VO constraints themselves, related to location, timeliness etc.

An effective approach to verify trustworthiness is based on trust negotiation (TN) protocols [9,15,16,21] which establish trust through the exchange and verification of parties' credentials.

These credentials typically encode properties, of different natures, that are deemed relevant by the VO in order to establish trust. TN protocols represent an effective means to bootstrap and manage trust relationships in dynamic VOs. Trust negotiations help in determining and verifying with a relatively small number of messages the properties, the history and, if needed, the reputation of VO's members. Further, by disclosing credentials, parties can learn about the capabilities of other members, or potential members, and establish whether or not they can provide the services needed for the success of the VO.

In this paper, we claim that semantic TN protocols can help in solving many issues related to trust management across VO members. In order to enable parties to establish trust within the context of VOs, we provide an integrated tool for VO Management that provides negotiation capabilities.

Specifically, we describe how we have extended different stages of the VO lifecycle by the introduction TN techniques. Our extended lifecycle is supported by the integration of the Trust-*X* TN system [15,16,21]. Trust-*X* is a lightweight TN tool offering a number of negotiation strategies catering to different levels of confidentiality that may be required by the negotiation parties. Additionally, Trust-*X* relies on the use of ontologies, which provide a clear semantics of the credentials employed during negotiations. The support of ontologies is crucial to ensure unambigous communication among negotiating parties, which in case of a VO may belong to different domains and may not share the same credentials' language. Finally, Trust-*X* is well

* Corresponding author.
E-mail addresses: asquicciarini@ist.psu.edu (A.C. Squicciarini), paci@disi.unitn.it (F. Paci), bertino@cerias.purdue.edu (E. Bertino).

0920-5489/$ – see front matter © 2010 Elsevier B.V. All rights reserved.
doi:10.1016/j.csi.2010.03.003

suited for short and efficient negotiations. Our discussion is based on an application scenario in the area of business oriented VOs. We show how the introduction of TN protocols does not result in significant latency in the VO operations.

The rest of the paper is organized as follows. In the next section we briefly illustrate the main steps of the VO lifecycle. Section 3 presents a running example that we will use in the paper for illustrative purposes. In Section 4 we present the trust negotiation paradigm and we discuss interesting aspects related to the semantics of trust negotiation. We present the trust establishment protocols in VO in Section 5, and describe the prototype integrating the TN in VO systems in Section 6. Interesting integration issues are discussed in Section 6.3. We present some relevant related work in Section 7, and conclude the paper in Section 8.

2. Virtual Organization Lifecycle

In order to discuss the role of TN protocols in VO Management, we summarize the main phases in a VO lifecycle. In the discussion we assume that the entities interested in participating in the VO are service providers (SP).

- *Preparation for participation in the VO*. This is a preliminary phase and reflects the necessary steps that a SP has to take in order to participate in the VO. SPs publish their resources' functionalities in a public repository. The resources' description provides detailed information about resources' capabilities, the resources' interaction means and other information like the resource quality. This information allows one to select a SP for inclusion in the VO.
- *Identification*. This phase is considered as the first major phase in the VO lifecycle and starts when an organization, referred to as VO Initiator, identifies a business goal and thus defines a contract to fulfill it. The contract states the roles and the requirements that each member has to fulfill in order to be part of the VO. In addition, the contract specifies the collaboration rules the VO members have to follow to reach the business goal.
- *Formation*. The VO Initiator queries public repositories to retrieve the information published during the Preparation phase. The Initiator uses such information to select a set of potential VO members that match the contract's requirements. The VO Initiator then sends them an invitation to join the VO containing the terms of the contract they have to fulfill. If they accept the invitation, they become members of the VO. Each member will have an associated reputation, established on the basis of past transactions and updated as it interacts with members of the VO.
- *Operation*. Once the VO is set up, its members cooperate according to the collaboration rules specified in the contract. The operation phase has several critical security issues. VO members may exploit their privileges and misuse the resources available, gather information about other enterprises for personal gain or fail to fulfill the contract rules, and even take advantage of the resources made available to perpetrate crimes. All the interactions must be monitored, ruled by security policies and any violation must be notified. Reputation of the members is updated accordingly based on the result of the operations, the quality of the service granted and so forth. If a VO member violates the contract, it can either be replaced or it can be punished; for example its reputation can be negatively modified.
- *Dissolution*. This phase takes place when the objectives of the VO have been fulfilled. The VO structure is dissolved and final operations are performed to nullify all contractual binding of the VO's members.

3. Running example

An aircraft company is a prime contractor for an aerospace project developing a civil aircraft. Due to environmental regulations and rising fuel costs the aircraft must have low emissions and efficient fuel consumption. To meet this requirement the prime contractor decides to create a VO of smaller companies that provide services offering the required design/analysis capabilities. The VO is formed by the following members: a) the aircraft company that initiates the Aircraft Optimization process; b) an aerospace company that provides an engineering web portal (Design Partner Web Portal) that hosts an industry-standard product design database; c) a scientific/engineering consultancy that has developed an advanced optimization capability specifically for aerospace design work (Design Optimization Partner Service); d) a High Performance Computing Service provider (HPC Partner Service) to perform numerical simulations; and e) a storage provider (Storage Partner Service) for storing industrial engineering analysis data. Fig. 1 shows the interactions among the VO members during the operational phase of the VO. The dashed lines represent the TN protocols interleaving with the VO formation phase (arrow 0) and operation phase (arrow 3a).

The Aircraft Company's engineer selects a wing design by the Design Web Portal. The engineer decides to optimize the design. The Design Optimization Partner Service is first activated and then accesses the design-optimization control file from the Design Partner Web Portal. The file is sent to the HPC Partner Service which computes a new wing profile and computes a flow solution, generating new wing lift and drag values which are stored at the storage provider service. This data is then used to compute a revised design. Note that these steps (Steps 5 and 6) are executed repeatedly until the target result is achieved.

4. Trust negotiations and the Trust-*X* system

Trust negotiation (TN) is an authorization mechanism for open systems, in which interactions occur among parties that may be unknown to each other [9,13,15,21]. The goal of TN protocols is to enable parties to establish some level of mutual trust to exchange sensitive information and/or resources.

In this section we present the TN features by means of Trust-*X* [16], a comprehensive XML-based framework for trust negotiations specifically conceived for peer-to-peer environments. We first give an overview of *X*-TNL, the negotiation language supported by Trust-*X*. Subsequently, we discuss how we have enriched Trust-*X* with ontologies, so as to provide an expressive language supporting policies at different levels of granularity and enabling different VO parties to negotiate despite their possible usage of different credentials and policy languages. Finally, we provide an overview of Trust-*X* negotiation protocol.

4.1. Trust-X language

X-TNL is the XML-based language developed to specify information required to carry on trust negotiations, namely credentials and disclosure policies. *X*-TNL credentials are the means to convey information about the profile of the parties involved in the negotiation. A credential is a set of identity attributes of a party issued by a Credential Authority (CA). All credentials associated with a party are collected into a unique XML document, referred to as *X-Profile*. The disclosure policies state the conditions under which a resource or a credential can be released during a negotiation. Conditions are expressed as constraints on the attribute credentials owned by the parties involved in the negotiation. Each party adopts its own Trust-*X* set of disclosure policies to regulate release of local information (that is, credentials or policies) and access to services.

Like credentials, disclosure policies are encoded using XML. Regardless of the specific implementation, disclosure policies can be modeled as logic rules. Two building blocks for specifying disclosure policies are *terms* and *R-Terms*. A term is an expression of form $P(C)$ where P is a credential type and C is a (possibly empty) list of conditions

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

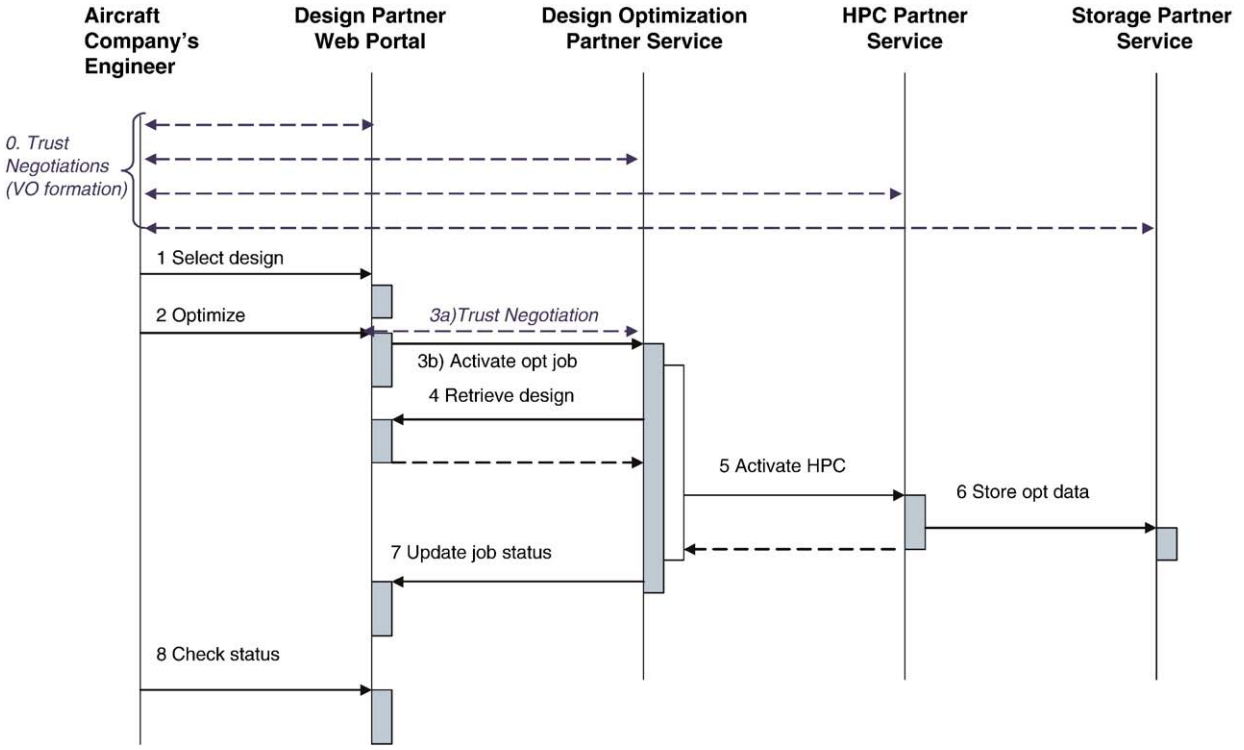

Fig. 1. Collaborations in the Aircraft Optimization VO.

on the attributes encoded in credentials of type *P*. The credential type *P* can be unspecified (and denoted by a variable), so to express constraints on the counterpart properties without specifying from which types of credential such properties should be obtained from. Such an approach gives the receiver of the policy the flexibility of choosing which credentials to send as a proof of policy satisfiability. *R-Terms* are expressions of the form *ResName(attrset)* where *ResName* denotes a resource name whereas *attrset* denotes a set of attributes, specifying relevant characteristics of the resource. Examples of resources are a credential, a file or a Web service.

As such, disclosure policies can assume one of the following forms:

1) $\mathbf{R} \leftarrow \mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n, n \geq 1$, where $\mathcal{T}_1, \mathcal{T}_2, \ldots, \mathcal{T}_n$ are terms and **R** is an *R-Term* identifying the name of the target resource.
2) $\mathbf{R} \leftarrow$ **DELIV**. A rule of this form is called *delivery rule*, meaning that *R* can be delivered as is.

A disclosure policy is *satisfied* if the stated credentials are disclosed to the policy sender and the policy conditions (if any) evaluated as true, according to the specific credential content. A delivery rule implies that the resource *R* is ready to be released, and no specific requirement has to be satisfied.

Example 1. The following are examples of disclosure policies:

- *VoMembership* ← *WebDesignerQuality*
- *QualityCertification* ← *AAACreditation*.

The first policy states that in order to obtain the VO membership a party has to prove that quality compliance with the Web Designer to regulations. In order to give proof of the compliance to quality a party has to prove that has an accreditation released by the American Aircraft associations.

4.2. Trust-X negotiation process

Trust negotiation is the approach to establish a mutual trust relationship between two parties that do not know each other and want to exchange on line resources or services. Trust is established through an exchange of digital credentials. Such credentials are identified during the negotiation process, within which each party decides which credential is willing to disclose to the counterpart and under which conditions.

In Trust-χ, the negotiation is performed in two main phases: the *policy evaluation phase* and the *credential exchange phase*. The key phase of a Trust-χ negotiation is the *policy evaluation phase*, which consists of a bilateral and ordered policy exchange. The goal is to determine a sequence of credentials, called *trust sequence*, satisfying the disclosure policies of both parties.

During each interaction, one of the two parties sends a set of disclosure policies to the other. The receiving party verifies whether its χ-*Profile* satisfies the conditions stated by the policies, and whether there are local policies regulating the disclosure of the credentials requested by the policies sent by the other party. If this is the case, the receiving party sends to the other party the disclosure policies protecting the credentials requested by the other party. Otherwise, the receiver informs the other party that it does not possess the requested credentials. The counterpart then sends an alternative policy, if any, or halts the process, if no other policies can be found. The interplay goes on until one or more potential trust sequences are determined, that is, whenever both parties determine one or more sets of policies that can be satisfied for all the involved resources. To maintain the progress of a negotiation and help detecting a potential trust sequence a tree structure is used. The trust sequence is identified by one tree view, where a view denotes a possible trust sequence that can lead to the negotiation success. The view keeps track of which terms may need to be disclosed to contribute to the success of the negotiation, and of the correct order of certificate exchange. More precisely, a negotiation tree is a labeled tree rooted at the resource that initially started the negotiation. Each node corresponds to a term, whereas edges correspond to policy rules. A negotiation tree is characterized by two different kinds of edges: simple edges and multiedges. A simple edge denotes a policy having only one term on the left side component of the rule. By contrast, a multiedge links several simple

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

edges to represent policy rules having more than one term on their left side component. Nodes belonging to a multiedge are thus considered as a whole during the negotiation.

Example 2. Fig. 2 represents an example of negotiation tree. The tree represents the negotiation between the Aerospace company that requests the release of a VO Membership certificate to the Aircraft company in the scenario of Section 3. The Aircraft company sends to the Aerospace company a policy for the release of the membership. The Aerospace company has to prove that meets the quality compliance regulations to the Web Designer. The Aerospace company, in order to give proof of the compliance to quality, wants the Aircraft company to prove that has an accreditation released by the American Aircraft associations, or to disclose a recent balance sheet.

Once the parties have agreed on a trust sequence, the *credential exchange phase* begins. Each party discloses its credentials, following the order defined in the trust sequence, eventually retrieving those credentials that are not immediately available through credentials chains. Upon receiving a credential, the counterpart verifies the satisfaction of the associated policies, checks for revocation and validity dates, and authenticates the ownership (for credentials). The receiving party then replies with an acknowledgment, and asks for the subsequent credential in the sequence, if any. Otherwise, a credential belonging to the subsequent set of credentials in the trust sequence is sent. The process ends with the disclosure of the requested resource or, if any unforeseen event happens, an interruption. If the failure is related to trust, for example a party uses a revoked certificate, the negotiation fails.

4.3. Ontologies in Trust-χ

Trust-χ, and more in general TN systems, builds on the assumption that parties are able to express trust requirements by means of disclosure policies, and that in specifying such requests they are aware of the credentials the counterpart could provide. Also, the underlying assumption is that parties have a common understanding of the type of credentials supported, and know their internal structure. Finally, another underlying yet important assumption is that credentials' semantics is clear and unambiguous.

In a VO setting, these assumptions may be too strong, due to the dynamic and heterogeneous nature of the parties joining a VO. Such parties may not be willing or able to express policies compliant with the restrictive TN credentials' representation, but require instead to be able to express a high level specification of such requests.

Ultimately, these assumptions are required due to the lack of a clear semantics of credentials and in general of the trust negotiation language. An interesting approach to address such issues is to employ ontologies and/or dictionaries [10] when conducting trust negotiations. In the context of the Semantic Web, ontologies provide a formal specification of concepts and their interrelationships, and play an important role in complex Web services environments, semantics-based search engines and digital libraries. Dictionaries have a more limited scope, but they are similar to ontologies, in that they provide a way to disambiguate similar names and assign a clear semantics to these names. Within Trust-χ and VOs, semantic enrichment makes it possible for TN parties:

- to express high level trust requirements;
- to map the trust requirement expressed by means of keywords onto lower level credentials.
- to employ local naming schemas, without worrying about mapping issues.

Trust-χ has thus been extended with a reasoning engine to facilitate its usage within VO. The engine relies on a reference ontology, capturing the main concepts used by the negotiation parties. We assume that each party maintains a local ontology and adds more concepts to it as needed. Precisely, the parties maintain an ontology modeling concepts and related instances in terms of credentials and attributes in the domain of interest. Each concept in the ontology is associated with the concept name, a set of attributes and credential types names. ⟨gender; Passport.gender; DrivingLicense.sex⟩ is an example of concept.

In this case, the concept known as gender can be implemented by the attribute Passport.gender or the attribute DrivingLicense.sex. In other words, a concept can be implemented by attributes of different credentials or by different credentials.

Within the ontology, concepts are related by different relationships, and hierarchically organized according to the conventional *is_a* relationship. As such, if concept C_i is in a relation *is_a* with C_k, the information conveyed by concept C_i can be used to infer information conveyed by concept C_k. For instance, the concept Texas_Driver License *is_a* Civilian_Driver License, since if an individual has a driver's license issued in Texas, then he/she has a civilian license.

The use of the ontology enables policy specification by means of list of concepts, and conditions (added within brackets at the end of the policy) against the concepts of interest, if any.

The party receiving such policy can easily infer the intention of the counterpart, and part of its business strategy. This is particularly the case in the context of VOs, where parties are often companies related by business contracts. To avoid full disclosure of such sensitive information, one may specify higher level policy conditions, expressed at a conceptual level. Such concepts can then be mapped onto specific credentials to be disclosed with the help of the reference ontology. This approach will free the negotiator from the burden of knowing the credentials' syntax and it will help hiding specific information that the business partner is trying to gather from the policy.

By expressing the policy through concepts, the VO party can avoid having to request a specific Id type and, for example, verify that the counterpart has an Intel issued card at run time without revealing that this is the one kind needed. Additionally, it can ask for a generic business list, rather than naming exactly the type of document needed to satisfy the request of showing some business proof.

Moreover, an interesting issue to address while integrating TN within VO is possible naming issues. The extension of Trust-χ with the reasoning engine facilitates the interoperability among the negotiation parties, by bridging the potential semantic gaps resulting from the usage of different naming schemas.

4.3.1. Using ontologies within trust negotiations

Ontologies are used both when defining disclosure policies and when interpreting a counterpart's policy. In the first case, the disclosure policies' can be abstracted by executing a substitution operation of sensitive credentials names into the associated concepts names, which are more generic and disclose less information. The process can be iterated so as to hide even more information, if the ancestor concept is used.

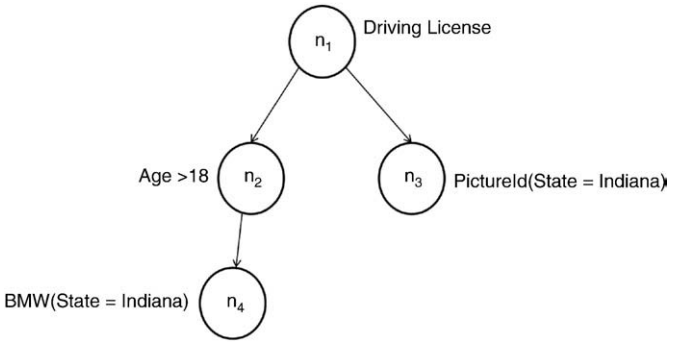

Fig. 2. Simple example of negotiation tree.

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

Regarding the second case, Algorithm 1 reports a simple algorithm representing how disclosure policies are expressed by means of concepts. Given a certain policy, expressed in terms of concepts and related conditions over them, the algorithm first searches the required concept in the local ontology. If the concept does not belong to the ontology, a similar concept is determined in the local ontology, by using the similarity matching algorithm (lines 20–29). Once the concept of interest is identified, the algorithm determines the corresponding credential to be sent to the counterpart. In case more than one credential is available for the identified concept, the selection occurs based on the credentials' ownership (obviously the credential must be readily available) and its sensitivity. Sensitivity is by assumption represented by means of a label associated with each credential, and it can be determined efficiently in an automated fashion. The label takes values from the set {*low, medium high*}. In order to identify the set of local credentials corresponding to a certain sensitivity level, we employ the function *CredCluster* (lines 5), which clusters the credentials into groups,

Algorithm 1. Mapping algorithm

Require: Reference ontology ⟨CSet,R⟩, *pol*← C_1,...,C_k, disclosure policy to be locally satisfied.

```
1: for C_i, i ∈ [1, n] do
2:   while Flag = false do
3:     if C_i ∈ CSet then
4:       Let Cred_1, Cred_k be the credentials associated to C_i
5:       LowCredCluster(Cred_1,..., Cred_k, low)
6:       if LowCredCluster ≠ ∅ then
7:         Return Cred_j {Let Cred_j be a local credential ∈
           LowCredCluster}
8:       else
9:         LowCredCluster(Cred_1,...,Cred_k, med)
10:        if MedCredCluster = ∅ then
11:          return Cred_j {Let Cred_j be a local credential ∈
             MedCredCluster}
12:          HighCredCluster(Cred_1,..., Cred_k, high)
13:          if HighCredCluster = ∅ then
14:            return Cred_j {Let Cred_j be a local credential ∈
               HighCredCluster}
15:            Flag=True
16:          end if
17:        end if
18:      end if
19:    else
20:      Confidence=1;
21:      for C′ ∈ Cset do
22:        Sim = ComputeSimilarity(C′,C_i)
23:        if Sim > Confidence then
24:          Confidence=Sim
25:        end if
26:      end for
27:    end if
28:  end while
29: end for
```

according to their privacy label. In case the concept is not found in the local ontology, the negotiator needs to identify the local concept that better matches the required one (lines 20–29).

Semantic variations may in fact arise when attribute names or categorical values are associated with different ontologies.

The negotiator, when receiving a request expressed by means of concepts that are not included in the local ontologies can compute the mapping according to a matching algorithm, and resolve the ambiguity. In general, given ontologies O_1 and O_2, an ontology matching algorithm [1] takes O_1 and O_2 as input and returns a mapping $MO_1 \leftarrow O_2$ between the two ontologies. The mapping $MO_1 \leftarrow O_2$ contains for each concept (node) C_i in ontology O_1 (O_2) a matching concept C_j in O_2 (O_1) along with a confidence measure m, that is, a value between 0 and 1, indicating the similarity between the matched concepts. For our purposes, we require only the matching value of the concept C appearing in the counterpart policy with the concepts belonging to the ontology O_2. The concept with higher similarity score is the one selected. This is achieved by taking C and matching it with every concept in ontology O_2. The matching operation is executed according to the Jaccard coefficient [2], as developed for the GLUE mapping tool, and is summarized by the *ComputeSimilarity* function in Algorithm 1.

5. How to establish trust in a VO

We now elaborate on the integration of TN within the various phases of the VO lifecycle. Then we show through an application scenario examples of TN executed within a VO.

5.1. TN in the VO lifecycle

As sketched in Fig. 3, there are three main interaction points between VOs and trust negotiations:

- *Identification phase*. The VO Initiator, in addition to what discussed in Section 2, locally defines the disclosure policies to be used during the TN with potential members. Policies are created for the specific VO and in particular for the roles the VO potential members will play in the VO. They can require for instance credentials certifying the quality of service of the potential VO members, or tickets attesting their participation to other VOs.
- *Formation phase*. The VO Initiator engages a TN with the potential members accepting its invitation. The VO Initiator may engage multiple negotiations for a same role, to ensure that the role will be covered by at least one member (see Fig. 4). The potential members may specify disclosure policies either beforehand or on the fly before starting the TN. Examples of policies created in advance are the ones protecting sensitive credentials, while transient policies are specific to the VO. Disclosure policies of this kind may check the VO Initiator affiliation, its external partners (if any), and other possible VO properties that were not advertised. At the end of the successful negotiation both negotiators can decide whether or not to proceed further. Note that unlike the conventional joining phase of a VO, acceptance in TN is mutual: the potential member can decide to join the VO based on what it learns about the VO Initiator and the VO goal, that can in turn decide to choose or not that particular organization to provide the service. If the VO Initiator decides to assign the VO potential member to the role, it sends it a VO membership certificate that the member can use to identify itself during the operational phase. If a negotiation is not successful, the VO Initiator removes the invited VO partner from the potential partners list and looks for other potential members who have received the invitation.
- *Operational phase*. TN protocols are also useful in case of long lasting VOs, where credentials used for the VO formation may expire or be revoked before the VO dissolution. As an example, consider a quality of service certification which needs to be renewed on a yearly basis. Moreover, in complex VOs, the VO's activities may not all be described by a sequence of operations, and thus the operations' flow may change as the VO evolves. In these cases the VO's members may need additional information about partners' trustworthiness. This is the case, for example, of VO formed in grid computing, which involve very complex collaborations among the members. Unlike TN carried out during the formation phase, the result of a TN, in this case is not a credential, but it is an authorization to execute the next VO operations. A failed TN may compromise the VO lifecycle if the missing trust was fundamental for the continuation of the VO operations. Like the successful case, the failed TN may affect the

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

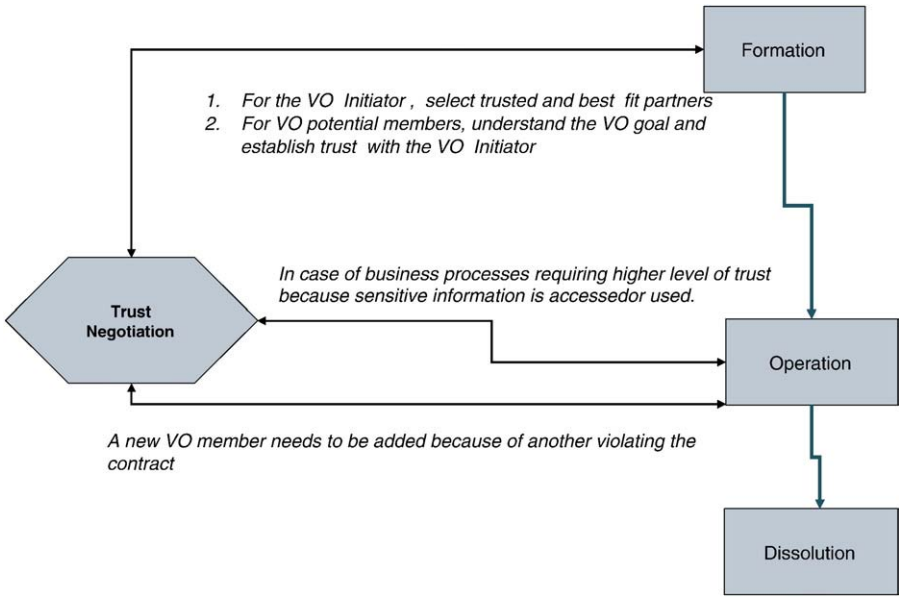

Fig. 3. Interleaving of trust negotiation in VO lifecycle.

parties' reputation. A TN is also executed in case of a VO member replacement by following the same protocols of the formation phase (third arrow of Fig. 3).

Examples of TN carried out in this scenario are:

- When receiving the invitation the Aerospace Company engages in a TN with the Aircraft Company's engineer. It requests the release of a VO membership certificate for that Aircraft Optimization VO. In a nutshell:
  - The Aircraft Company in order to release the VO membership certificate requires a certification from the Aerospace Company providing the Design Partner Web Portal. The required certificate should prove that the design processes that can be activated through the web portal are compliant with the UNI EN ISO 9000 regulations. The specified policy has the form *VoMembership* ← *WebDesignerQuality*, {UNI EN ISO 9000}, that is, the Aircraft Company will release the VO membership credential if the Web Portal proves the quality compliance the Web Designer regulations.
  - The local trust negotiation agent of the Aerospace Company maps the request into local credential *Credential* that is associated with the concept expresses by the counterpart policy. Second, it asks to verify the Aircraft Company accreditation released by the

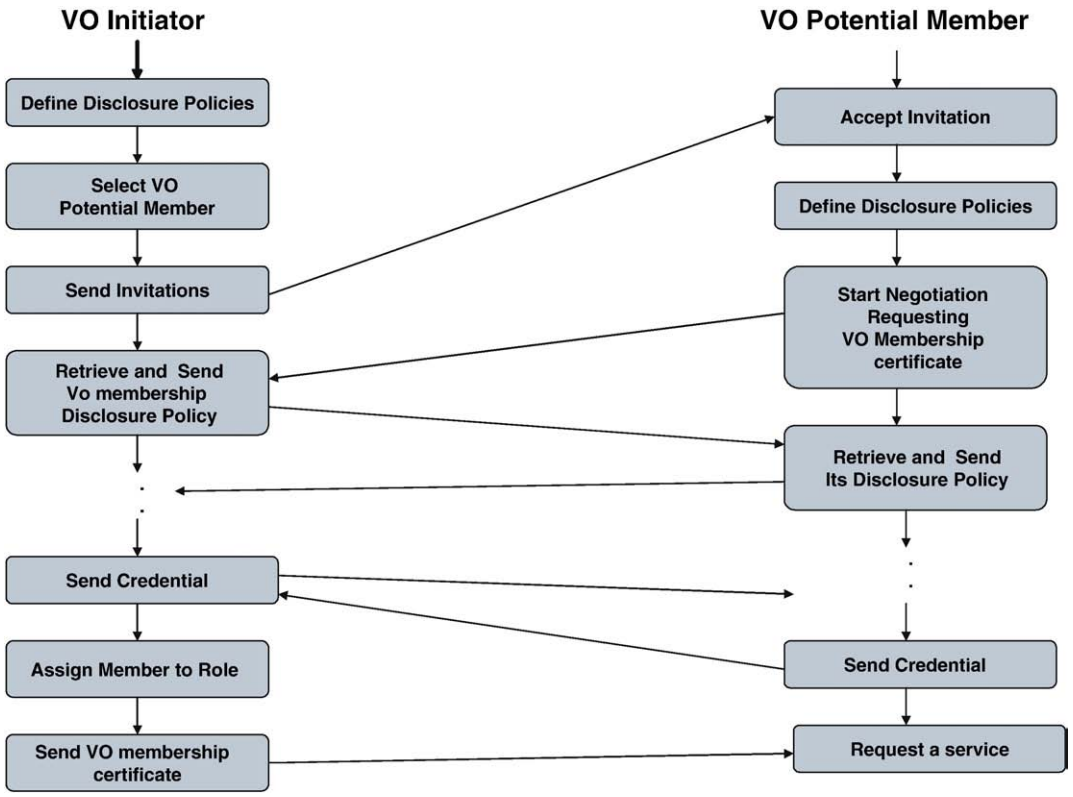

Fig. 4. TN between the VO Initiator and a VO potential member in the formation phase.

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

American Aircraft associations. The policy sent to the Aircraft Company's engineer is *Certification()←AAAccreditation()*.
- Since it only collaborates with well-established companies, the Aerospace Company also adds the following policy asking for the most recent balance sheet issued by a known certification company: *Certification()←BalanceSheet*.
- The Aircraft Company's engineer maps also the last request into a corresponding low sensitive credential. Suppose that the BalanceSheet corresponds to the credential *CertificationAuthorityCompany* (*Issuer = BBB*). Once credentials' mapping is completed by the reasoning engine of the two Trust-*X* agents, the company provides the required the credentials and so does the Aerospace company. Credentials are successfully verified using credential issuers' public keys.
- Aircraft Company decides to choose the Aerospace Company as provider of the Design Web Portal and releases the membership token to the Aerospace Company. The membership token contains the public key of the VO to be used for authentication in the VO.

- The design optimization partner needs additional information from the design web portal to check that the ISO 002 certificate is still valid, as a few months are passed after the VO formation. Here, a TN is executed to ensure that this certification is still valid. Additionally, as the new certificate has some confidentiality requirements, some additional policies will be involved. Specifically, the policy will be satisfied upon disclosure of the requester of the same certification. The requirements of the two parties are to prove the compliance with privacy regulation. At the lower level, the policies to be satisfied are: *Certification()←PrivacyRegulator()* and *PrivacyRegulator()← PrivacyRegulator()* in response to the Aircraft Company one.
- During the operational phase one of the members detects that the reputation of the HPC service has decreased due to contract's violation. This information is sent to the Aircraft Company's engineer that selects a new HPC service provider. The new member is enrolled, using a TN.

6. System architecture and implementation

The core of the system architecture is the VO Management toolkit which provides services and protocols for the VO lifecycle phases. In order to support trust negotiations during the VO Formation and VO Operation phases, we have integrated into the VO Management toolkit a new Web Service that implements TN protocols. In Fig. 5, we report a graphic representation of the overall architecture. As shown, the TN system is integrated as part of the VO Management tool, and invoked as a web service when needed. Upon invoking the TN system, the TN process is activated, and its additional services used according to the negotiation's evolvement. In what follows we describe the VO Management tool and the TN Web service; we then describe how the TN service has been integrated into the VO Management tool.

6.1. VO Management toolkit

The VO Management toolkit is a Web-based application, developed by SAP [12] in the TRUSTCOM European project [17], built over a SOA [20] combining several Web services for managing VOs. The VO Management toolkit has been written in Java and provides a unified GUI supporting the administrator to easily handle all aspects of VO Management. The toolkit is deployed as three distinct components:

- The Host Edition provides services such as member registration and VO monitoring. It shows the active VO and the list of services that are available for participating in a VO (this includes the ones that are already in a VO plus the ones that are waiting for an invitation).
- The Initiator Edition allows one to create a VO. It provides the user interface to all services required for managing a VO.
- The Member Edition allows the participation in a VO. It registers itself in a Host, and allows members to configure properties and send/receive e-mails.

Here, we shortly describe how the Initiator Edition implements the discovery of the potential VO members and invites them to cover a particular role in the VO. The Initiator Edition is equipped with a user interface which displays all the potential members. Potential members are identified based on the roles that they have registered for. One or more members can be selected to play the roles to be covered in the new VO. Invitations appear in the Mailbox of the new potential members. The message contains the text entered in the invitation screen. When all the members have accepted the invitation, the “Role overview” screen shows the possible members that can be assigned to each role by clicking the “Assign Member” button.

6.2. The TN service

The TN Web service supports the operations to carry on a TN according to the standard, the strong suspicious, the suspicious and the trusting negotiation strategies (Section 3). It has been developed in Java, using the Tomcat Application Server, the Axis Soap Engine, and Oracle version 10g. The TN Web service provides three different operations, *StartNegotiation*, *PolicyExchange* and *CredentialExchange*, each corresponding to one of the main phases of the negotiation process.

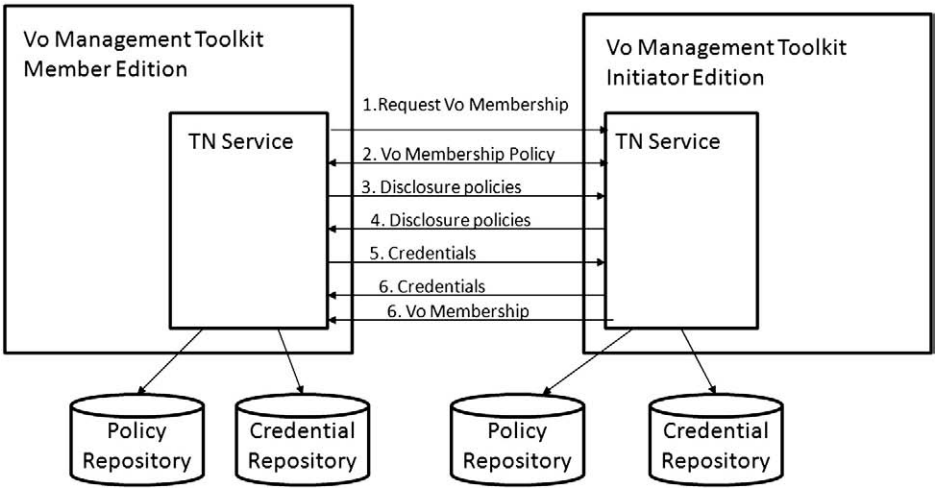

Fig. 5. System architecture.

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

StartNegotiation has, as input message, a StartNegotiationRequest, that specifies the negotiation strategy selected by the party who invokes the operation, the URL of the counter party in the negotiation process and the parameters to connect to the Oracledatabase containing the disclosure policies and credentials of the invoker. StartNegotiation assigns a unique id to the negotiation process and opens the connection with Oracle database. The negotiation id is returned in the output message StartNegotiationResponse. PolicyExchange checks if the database contains disclosure policies protecting the credentials requested in the counterpart's disclosure policies, which are listed in the message PolicyExchangeRequest. If this is the case, they are inserted in the response message PolicyExchangeResponse. CredentialExchange receives as input the message CredentialExchangeRequest and verifies the validity of the counterpart's credential contained in the message. Then, it selects the next credential to be sent to the negotiation partner. The selected credential is returned in the message CredentialExchangeResponse. The tool supports an XML proprietary format to represent credential and disclosure policies. In Example 1 we show how credentials and disclosure policies can be represented in the proprietary XML format.

A client application has also been developed, ClientWS.java, implementing the negotiation protocol by invoking the Web service's operations. It provides a GUI, by means of which users specify the parameters of the negotiation and enable them to monitor the negotiation process.

Example 1. A credential consists of three main subelements: the <*header*>, the <*content*>, and the <*signature*> elements. The <*header*> includes information like the credential type (<*credType*> element), the issuer (<*issuer*> element), and the time frame validity of the credential (<expiration _Date> element). The <*content*> element contains all the attributes that characterize the credential type. The <*signature*> includes the signature of the issuer on the whole credential encoded in base64. Fig. 6 illustrates an example of credential "ISO 9000 Certified" that the Airspace Company offering the Design Web Portal Service can provide during a negotiation process with the Aircraft Company's engineer. The credential "ISO 9000 Certified" is of type "ISO 9000 Certified" and has been issued by INFN certification authority with validity from the 2009-10-26T21:32:52 to the 2010-10-26T21:32:52. The credential "ISO 9000 Certified" has only one attribute called "QualityRegulation"specifies the quality regulations to which the Design Web Portal Service is compliant.

A policy is defined essentially by three components: <*resource*>, <*properties*> and type. The <*resource*> element simply specifies the credential protected by the disclosure policy (target attribute). The <*properties*> element specifies the conditions that the credential of the other party should satisfy for the release of the credential. This element has as many subelements, named <*certificate*>, as the number of conditions. The element <*certificate*> has an attribute named "targetCertType", denoting the credential type on which the condition is specified. Additional conditions to be evaluated on the credential attributes are specified in the subelements <*certCond*>. Such element stores an Xpath expression on the credential denoted by "targetCertType". If no <*certCond*> elements are specified, it means that the policy applies to all the subjects qualified by "targetCertType". Fig. 7 represents the disclosure policy for the credential "ISO 9000 Certified" defined by the Aereospace Company. The policy specifies that for disclosing the "ISO 9000 Certified" credential, the Aereospace Company wants that the AirCraft Company presents an Aircraft Company accreditation credential released by the American Aircraft associations.

6.3. Integration

The integration of the TN Web service with the VO Management toolkit was greatly simplified by the fact that both are implemented using a SOA. Despite that, it was necessary to modify both tools in order to integrate them.

The major integration issues we had to face were related to the credentials and disclosure policies management, from their format to their storage. Trust-*X* uses attribute credentials encoded in a proprietary XML format; while the VO Management toolkit supports X.509 [18] identity credentials to identify the VO members during the VO operational phase. We thus upgraded the TN Web service in order to support both our XML proprietary format and the X.509 v2 format for attribute certificates. Then, we modified the TN service code to allow the VO Initiator to create at runtime the VO membership credential: this is an X509 credential that is released to the VO member when it is assigned a VO role. Once modified the TN Web service's code we added it to the services already used to implement the VO Management toolkit.

A drawback of using X509 v2 credentials is that only the standard and trusting negotiation strategies can be adopted, because this standard does not support partial hiding of the credential contents. We are investigating techniques to address this shortcoming of the X509 v2 certificates in order to support selective disclosure of attributes. This would make it possible to support the suspicious and strong suspicious negotiation strategies with such credential format. One solution would be to substitute the attributes in clear with attributes whose content is the hash value of the concatenation of attribute name and attribute value. The signature could be computed over the whole hashed content. We are exploring the robustness and computational complexity of this approach. Hence, it has not been included yet in our prototype.

```
<credential>
       <header>
            <credType>ISO 9000 Certified</credType>
            <credID>1100</credID>
            <issuer HREF="http://www.ItalyCountry.com"
                          Title="KTHUUniversity_Repository" />
            <expiration_Date>
                <notBefore>2009-10-26T21:32:52</notBefore>
                <notAfter>2010-10-26T21:32:52</notAfter>
            </expiration_Date>
     </header>
    <content>
      <qualityRegulation Id="1">UNI EN ISO 9000</qualityRegulation >
    </content>
   <signature>
     ...
   </signature>
  </credential>
```

Fig. 6. Example of credential in XML format.

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

```
<policySpec >
  <properties >
     <certificate targetCertType=AAAccreditation>
     </certificate>
   </properties>
   <resource target= ISO 9000 Certified />
</policySpec >
```

Fig. 7. Example of disclosure policy in XML format.

Moreover, as discussed in Section 4.3 in the context of VOs we cannot assume that all the VO partners adopt the same naming system to specify their credentials and disclosure policy attributes. Hence, we have extended the Trust-*X* system with Jena [5] reasoning engine supporting ontologies. We have used OWL [10] to create a common ontology for the credential and disclosure policies attributes. Fig. 8 shows an extract of the OWL ontology for credential attributes.

In order to take advantage of OWL, we have also extended the TN service's PolicyExchange and CredentialExchange operations, so as to enable policies and credentials representation using high level concepts. When a VO partner sends to the counterpart, the ontology representing his/her disclosure policies, the PolicyExchange matches the concepts representing the attributes in the policy ontology with the one representing the credentials of the receiving VO partner. If there is no matching, the policy cannot be satisfied by the receiving party. The credential Exchange operation adopts ontology matching to verify that the credential sent by the counterpart matches the local policy ontology. To implement the ontology matching functions, we have used the Falcon-AO v 0.7 [3] API.

The VO Management toolkit adopts MySQL as storage support, while the TN service uses Oracle. Compared to Oracle, MySQL has very few features to support the storage of XML data and the execution of XPath queries on them. However, we migrated from Oracle to MySQL since during the integration we tried to minimize the VO toolkit changes, when possible.

```
<owl:Ontology rdf:about="XMLCredential">
  <rdfs:comment>An ontology for credentials</rdfs:comment>
  <rdfs:label>XML Credential Ontology</rdfs:label>
  <owl:Class rdf:ID="credential"/>
  <owl:Class rdf:ID="header"/>
  <owl:Class rdf:ID="content"/>
  <owl:Class rdf:ID="credType"/>
  <owl:Class rdf:ID="credId">
      <owl:equivalentClass rdf:resource="&X509Credential;serialNumber"/>
  </owl:Class>
  <owl:Class rdf:ID="issuer"/>
  <owl:Class rdf:ID="expirationDate">
      <owl:equivalentClass rdf:resource="&X509Credential;Validity"/>
   </owl:Class>
   <owl:Class rdf:ID="notBefore"/>
   <owl:Class rdf:ID="notAfter"/>
   <owl:Class rdf:ID="signature"/>
   <owl:ObjectProperty rdf:ID="hasheader">
     <rdfs:domain rdf:resource="#credential" />
     <rdfs:range  rdf:resource="#header" />
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="hascontent">
     <rdfs:domain rdf:resource="#credential" />
     <rdfs:range  rdf:resource="#content" />
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="hassignature">
    <rdfs:domain rdf:resource="#credential" />
    <rdfs:range  rdf:resource="#signature" />
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="hascredType">
     <rdfs:domain rdf:resource="#header" />
     <rdfs:range  rdf:resource="#credType" />
   </owl:ObjectProperty>
   <owl:ObjectProperty rdf:ID="hascredId">
     <rdfs:domain rdf:resource="#header" />
     <rdfs:range  rdf:resource="#credId" />
   </owl:ObjectProperty>
    <owl:ObjectProperty rdf:ID="hasissuer">
    <rdfs:domain rdf:resource="#header" />
    <rdfs:range  rdf:resource="#issuer" />
  </owl:ObjectProperty>
  <owl:ObjectProperty rdf:ID="hasExpirationDate">
    <rdfs:domain rdf:resource="#header" />
    <rdfs:range  rdf:resource="#ExpirationDate" />
  </owl:ObjectProperty>
  ...
</Ontology>
```

Fig. 8. Sketch of the OWL ontology to match different credential formats.

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

In order to allow the storing of the ontologies expressed in OWL we have adopted the Core Protégé APIs [11] which allow one to store ontologies in different formats such as XML Schema.

To finalize the integration we had to link the VO Initiator and VO member editions web pages to allow the execution of TN related operations, such as the specification of disclosure policies and the negotiation process. We modified the code of the VO Initiator and the VO member editions with the creation of the Java class which generates (a) the jsp pages that allow one to define the disclosure policies and (b) the jsp page that triggers the negotiation and allows one to monitor it.

6.3.1. Experimental results

We present the results of our testing which consisted of timing the execution of the join process. The tests have been executed on a Pentium 4 PC with 2.00 GHz processor and with 512 MB of RAM, under Microsoft Windows XP. The performance has been measured in terms of CPU time (in milliseconds). Fig. 9 reports the experimental results. All tests were based on the Aircraft Optimization VO scenario. We measured the time of the airspace company offering a Design Partner Web Portal for:

a) Executing the join along with the negotiation with the Aircraft Company's. During the negotiation process the Airspace Company to join the VO has to provide ISO 9000 Certified and American Airspace Association Member certificates to the Aircraft Company's engineer, that in turn, has to provide the certificates American Aircraft Association Member and VO Membership, that proves that the airspace company is eligible to join the VO (label Join with trust negotiation).
b) Executing the join without the negotiation process (label Join).
c) Executing a TN (identical to the one in case (a)) from the standalone TN Web Service (label trust negotiation).

The time of the join process before the integration with the TN Web service (test b) is around 3 s while the time of the join process with TN (test a) is around 4 s. Therefore, the join process execution time only increases of 27 s.

7. Related work

Trust negotiation for web-based applications has been recognized as an interesting and challenging research area to explore, and has been extensively investigated in recent years. As a result, a variety of techniques and prototypes have been developed [14,19,22–24].

While much work has been conducted into the foundations of trust negotiation such languages (e.g.,[16,24,25]) protocols and strategies for conducting trust negotiations (e.g., [16,19,21]), and privacy issues in negotiations (e.g., [13,14,22,23]) only a few research groups have investigated the applications of trust negotiation within the Web services domain.

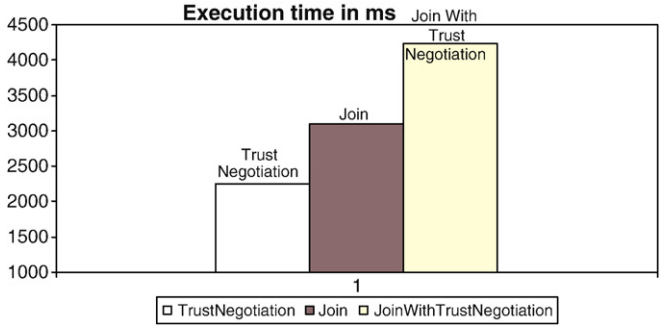

Fig. 9. Join execution times.

Concerning system architectures for trust negotiation, Hess et al. [4] proposed a trust negotiation in TLS (TNT) handshake protocol by adding trust negotiation features. Winslett et al. [24] proposed the TrustBuilder architecture for trust negotiation systems. The TrustBuilder architecture includes a credential verification module, a policy compliance checker, and a negotiation strategy module, which is the core of the system. The same group of researchers has more recently released Traust [8], which leverages the former TrustBuilder and acts as an agent between a browser and a portal acting as a service provider offering the trust negotiation service. Such prototypes, although interesting, cannot easily be integrated into the VO toolkit, due to their architectural design and the type of strategies supported. For example, Traust is supported as a proxy between a web-based browser and an agent. The VO toolkit does not interface with a web portal, therefore major architectural changes are required to integrate Traust into VOs. Additionally, TrustBuilder is a powerful system which level of sophistication and complexity goes beyond the needs of a VO. In [6], Koshutanski and Massacci describe a trust negotiation framework designed for Web services. This framework facilitates the composition of access policies across the constituent pieces of a workflow, the discovery of credentials needed to satisfy these policies, the management of the distributed access control process, and the logic to determine what missing credentials must be located and provided to satisfy a given policy. This work operates at the business process level by determining and satisfying the composite access control policy for a workflow prior to its execution. Furthermore, policies are represented using a datalog-based language. Therefore, even for this framework, integration would require major architectural modifications both at language and at the system level, resulting in a cumbersome integration process. Recently, an interesting contribution toward standardized trust negotiation systems, has been proposed by Lee and Winslett [7].

The authors showed that WS-Policy and WS-SecurityPolicy standards can be used to define a range of expressive trust negotiation policies. Also, they showed that the WS-Trusts can be extended to act as a standards-compliant transport mechanism within which trust negotiation sessions can occur. The authors also provide a proof-of-concept implementation of the TrustBuilder2 framework, showing how trust negotiation can be parameterized to act as a trust engine, as described by the WS-Trust standard. These types of extensions are desirable, in that the more standardized the TN system is, the easier it is to interface it with other SOA-based architectures. However, the TrustBuilder2 prototype is yet to be completed and not available, hence it is not possible to estimate the level of effort required for integration purposes with VO Management tools.

8. Conclusions

Virtual Organization represents a new collaboration paradigm in which the participating entities pool resources in order to achieve a common goal. Choosing the appropriate VO partners is a crucial aspect for the VO success. Ensuring trustworthiness of the members is fundamental for making the best decisions. In this paper, we have shown how TN represents an effective means to select the best possible members during different stages of the VO lifecycle. We discussed a concrete application scenario, and described the tools to integrate a well-known TN system, referred to as Trust-*X*, with a VO Management toolkit [].

The experimental results show that the integration of Trust-*X* into the VO Management toolkit is a price worth to pay because it elegantly extends the VO phases without significantly affecting the overall execution time. We plan to further extend this approach along two directions. A first extension is related to enhancing the Trust-*X* language to support the specification of policies with group conditions and requesting credentials that describe VO properties. A second extension is the support of XACML policies, which would make our

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003

integrated toolkit portable and interoperable with a number of other VO Management tools.

References

[1] N. Choi, I. Song, H. Han, A survey on ontology mapping, Proceedings of the *ACM SIGMOD International Conference on Management of Data*, Chicago, Illinois, USA, June 27–29, 2006.
[2] A. Doan, J. Madhavan, P. Domingo, A. Halevy, Learning to map between ontologies on the semantic web, Proceedings of the *Eleventh ACM International World Wide Web Conference*, 2002.
[3] Falcon-AO. The Falcon Project. Available at http://iws.seu.edu.cn/projects/matching/.
[4] A. Hess, J. Jacobson, H. Mills, R. Wamsley, K.E. Seamons, B. Smith, Advanced client/server authentication in TLS, *Proceedings of Network and Distributed System Security Symposium*, San Diego, CA, February 2002.
[5] Jena Ð A Semantic Web Framework for Java. Available at http://jena.sourceforge.net/.
[6] H. Koshutanski, F. Massacci, An interactive trust management and negotiation scheme, *Proceedings of the Second International Workshop on Formal Aspects in Security and Trust (FAST)*, 2004, pp. 139–152.
[7] A.J. Lee, M. Winslett, Towards standards-compliant trust negotiation for web services, Proceedings of the *Joint iTrust and PST Conferences on Privacy, Trust Management, and Security (IFIPTM 2008)*, June 2008.
[8] A.J. Lee, M. Winslett, J. Basney, V. Welch, The Traust Authorization Service, ACM TISSEC —Transactions on Systems and Information Security, vol. 11(1), 2008.
[9] L. Olson, M. Winslett, G. Tonti, N. Seeley, A. Uszok, J. Bradshaw, TrustBuilder as an authorization service for web services, Proceedings of *International Workshop on Security and Trust in Decentralized/Distributed Data Structures* (STD3S) Atlanta, Georgia, April 2006.
[10] D. McGuinness, van Harmelen, F. OWL Web Ontology Language Overview. Available at http://www.w3.org/TR/owl-features/.
[11] Protégé. Available at http://protege.stanford.edu/.
[12] SAP Business Software Solution Application Services. Available at http://www.sap.com.
[13] B. Sadighi Firozabadi, M. Sergot, Contractual access control, Proceedings of *IEEE Security Protocols, 10th International Workshop*, Cambridge, UK, 2002, pp. 96–103.
[14] K.E. Seamons, M. Winslett, T. Yu, Protecting privacy during on line trust negotiation, 2nd Workshop on Privacy Enhancing Technologies, San Francisco, CA, April 2002.
[15] I. Ray, A.C. Squicciarini, A.C.E. Bertino, E. Ferrari, Achieving privacy in trust negotiations with an ontology-based approach, IEEE Trans. Dependable Sec. Comput. 3 (1) (2006) 13–30.
[16] A.C. Squicciarini, E. Bertino, E. Ferrari, F. Paci, B. Thuraisingham, PP-Trust-*X* A system for privacy preserving trust negotiations, ACM TISSEC —Transactions on Systems and Information Security, vol. 10(3), July 2007.
[17] TrustCom European Project. TRUSTCOM, 2004. Available at www.eu-trustcom.com.
[18] X.509 Certificates RFC. Available at http://www.ietf.org/rfc/rfc2527.txt.
[19] T. Yu, M. Winslett, A unified scheme for resource protection in automated trust negotiation, Proceedings of *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2003, 16.
[20] Web Service Architecture W3C Working Group Note 11 February 2004 http://www.w3.org/TR/2004/NOTE-ws-arch-20040211/.
[21] W.H. Winsborough, K.E. Seamons, V.E. Jones, Automated trust negotiation, Proceedings of *DARPA Information Survivability Conference and Exposition*, Hilton Head, SC, January 2000.
[22] W.H. Winsborough, N. Li, Protecting sensitive attributes in automated trust negotiation, ACM Workshop on Privacy in the Electronic Society, November 2002.
[23] W.H. Winsborough, N. Li, Safety in automated trust negotiation, *IEEE Symposium on Security and Privacy*, Oakland, CA, May 2004.
[24] M. Winslett, T. Yu, K.E. Seamons, A. Hess, J. Jarvis, B. Smith, L. Yu, Negotiating trust on the Web, IEEE Internet Computing, vol. 6(6), November 2002, pp. 30–37.
[25] M. Winslett, C. Zhang, P. Bonatti, PeerAccess: a logic for distributed authorization, Proceedings of the *th ACM Conference on Computer and Communications Security (CCS 2005)*, 2005, pp. 168–179.

Elisa Bertino is Research Director of the Center for Education and Research in Information Assurance and Security (CERIAS). Previously she was a faculty member at the Department of Computer Science and Communication of the University of Milan where she was the Department Head and Director of the DB and SEC laboratory. She has been a visiting researcher at the IBM Research Laboratory (now Almaden) in San Jose, at the Microelectronics and Computer Technology Corporation, at Rutgers University, at Telcordia Technologies.
Her main research interests include security, privacy, digital identity management systems, database systems, distributed systems, and multimedia systems. In those areas, Prof. Bertino has published more than 400 papers in all major refereed journals, and in proceedings of international conferences and symposia. She is a co-author of the books "Object-Oriented Database Systems – Concepts and Architectures" 1993 (Addison-Wesley International Publ.), "Indexing Techniques for Advanced Database Systems" 1997 (Kluwer Academic Publishers), "Intelligent Database Systems" 2001 (Addison-Wesley International Publ.), and "Security for Web Services and Service Oriented Architectures" 2009 (Springer). She has been a co-editor in chief of the Very Large Database Systems (VLDB) Journal from 2001 to 2007. She serves, or has served, on the editorial boards of several scientific journals, including IEEE Internet Computing, IEEE Security and Privacy, IEEE Transactions on Knowledge and Data Engineering, ACM Transactions on Information and System Security, ACM Transactions on Web, Acta Informatica, the Parallel and Distributed Database Journal.

Anna Squicciarini is an assistant professor at the college of Information of Information Science and Technology, at the Pennsylvania State University. During the years 2006–2007 she was a post doctoral research associate at Purdue University. Squicciarini earned her Ph.D. in Computer Science from the University of Milan, Italy, in February 2006. During her Ph.D. she has been visiting scholar at the Computer Science Department of Purdue University, at the Colorado State University, and at the Swedish Institute of Computer Science. Squicciarini's main interests include access control for distributed systems, privacy, security for Web 2.0 technologies and grid computing. Squicciarini is an author or co-author of more than 40 papers in refereed journals, and in proceedings of international conferences and symposia.

Federica Paci is currently a post-doctoral at the Department of Engineering and Computer Science of the University of Trento. She works in the Secure Change European project. From February 2008 to March 2009 Dr. Federica Paci was a post-doctoral research associate at Purdue University. Paci's main research interests include access control for service oriented architectures, virtual organizations and trust negotiations. Currently, she is exploring security issues in the context of social networks and is developing trust negotiation protocols in peer-to-peer platforms. Paci earned her Ph.D. in Computer Science from the University of Milan, Italy, in February 2008. In February 2004, she received the equivalent of a combined bachelor's/master's degree in Computer Science, also from the University of Milan. During Spring Semester of 2005 and 2006, Paci was a visiting research scholar at CS Department of Purdue University, West Lafayette, IN. Paci an author or co-author of more than 20 conference papers and journal articles.

Please cite this article as: A.C. Squicciarini, et al., Trust establishment in the formation of Virtual Organizations, Comput. Stand. Interfaces (2010), doi:10.1016/j.csi.2010.03.003