Gestion de la confiance et intégration des exigences sociales au sein de communautés virtuelles

R. Yaich yaich@emse.fr
O. Boissier boissier@emse.fr
P. Jaillon jaillon@emse.fr
G. Picard picard@emse.fr

Ecole des Mines de Saint-Etienne, France

Résumé

La confiance est devenue un facteur clé des processus de décision au sein de communautés virtuelles. Le caractère ouvert et décentralisé de ces environnements couplés à leur dimension sociale défient les mécanismes actuelles de gestion de la confiance. Notamment pour ce qui concerne la gestion et l'intégration des exigences de confiance des utilisateurs et ceux de leurs communautés. Afin de répondre à ce problème, nous proposons SC-TMS, un système de gestion de la confiance adaptatif basé sur le principe de conformité sociale [4]. En nous appuyant sur les technologies multi-agents, des politiques de confiance sont utilisées pour spécifier à la fois les modèles de confiance centrés utilisateur et les modèles centrés communauté. Les agents y sont utilisés pour gérer et combiner ces différentes politiques de manière flexible et décentralisée. Nous décrivons les fonctionnalités et l'architecture qui les mettent en œuvre et discutons de leur implémentation.

Mots-clés : *Systèmes multi-agents, gestion de la confiance, politiques de confiance, communautés virtuelles*

Abstract

Recently, trust became a key factor in the decision process within virtual communities. The inherent decentralization and openness of such environments produced new challenges that are not solved by current approaches. In this paper we are considering how user-centred and community-centred trust policies can be considered, managed and combined. To this aim, we propose an adaptive Socially-Compliant Trust Management Systems (SC-TMS) based on multi-agent technologies. In this framework, trust policies can be used to specify (i) user-centred and (ii) community-centred trust models. Agents are used to manage and combine these different policies in a decentralized and flexible way. We describe the functionalities and the architecture that supports them and discuss also a prototype implementation.

Keywords: *Multi-agent systems, trust management, trust policies, virtual communities*

1 Introduction

Les environnements en ligne tels que les communautés virtuelles sont connues pour être dynamiques, incertains et risqués. Dans de tels contextes, où les mécanismes de sécurité standards ne sont plus utilisables, la confiance est rapidement devenue un facteur déterminant lors du processus de prise de décision. Cependant, plusieurs caractéristiques de ces communautés constituent une menace pour les mécanismes actuels pour l'établissement de cette confiance. A l'identification des facteurs clés d'un modèle de confiance efficace, s'ajoute le défi de trouver le formalisme approprié afin de représenter ces modèles.

Les politiques de confiance sont considérées comme une solution viable pour représenter les modèles de confiance [21]. Elles constituent un moyen pratique et flexible pour spécifier sous quelles conditions la confiance peut être accordée. Toutefois, un des principaux enjeux de la spécification, du déploiement et de l'application des politiques de confiance dans des communautés virtuelles ouvertes et décentralisées, est de faire face à l'imprévisibilité due à leur caractère dynamique et évolutif. Les changements imprévus doivent pouvoir être pris en charge au risque de voir l'exécution des politiques devenir rapidement inadaptées et inefficace. Un autre problème importante réside dans la capacité des politiques individuelles à s'intégrer de manière transparente aux exigences de confiance collectives qui peuvent être imposées ou recommandées par les communautés virtuelles elles mêmes. En effet, les humains, lors de leurs décisions usuelles, considèrent simultanément deux niveaux d'exigences : les *exigences collectives* qui représentent les critères de confiance communs qui devraient être utilisés par les membres d'une communauté, et les *exigences individuelles* qui reflètent des critères personnels [6, 16].

Ainsi, il devient de plus en plus évident que ni des modèles de confiance statiques ni des politiques de confiance inertes ne peuvent répondre

de manière appropriée à ces exigences. Que ce soit pour ajuster les politiques à des changements environnementaux ou pour permettre aux exigences individuelles d'être en accord avec les exigences collectives, enrichir les politiques de confiance avec des capacités d'adaptation est clairement nécessaire. Dans cet article, nous proposons de doter les systèmes de gestion de la confiance (**TMS**) [1] [5] de capacités d'adaptation basées sur un modèle de confiance riche, dynamique et intégratif. *Riche* signifie que la confiance est construite sur un large éventail de facteurs ; *dynamique* signifie que les politiques sont constamment adaptées et évoluent au cours du temps ; *intégratif* signifie que les politiques individuelles sont combinées avec les politiques collectives pour s'aligner sur les exigences de confiance de la communauté. A cette fin, nous proposons un modèle de gestion de la confiance basé sur des politiques qui sont considérées comme des implémentations concrètes des modèles de confiance [21].

La suite de l'article est structurée comme suit. La section 2 introduit les concepts de base sur la confiance et un scénario illustratif de notre problématique au sein de communauté virtuelle. La section 3 introduit les fondements de notre approche, incluant une vue générale du système et du modèle de confiance. La section 4 présente les détails de la représentation des politiques de confiance. L'architecture du *framework* est illustrée et discutée dans la section 5. Enfin, des remarques et une conclusion sont dressées en section 6.

2 Contexte et positionnement

L'intérêt pour la confiance n'a cessé de croître ces dernières années. Ainsi, la littérature sur le sujet est assez abondante et les définitions aussi diverses que variées. Dans cet article, nous ne souhaitons pas examiner ces définitions ni en proposer de nouvelles. Nous orientons le lecteur vers [3] qui résume la littérature existante. Cependant, compte tenu du caractère subjectif de la confiance, nous nous sommes attachés à l'identifier les facteurs (i.e. les sources d'information) essentiels pour sa construction. Concrètement, définir un *modèle de confiance* revient à préciser la nature (i.e. types), les valeurs (i.e. seuils) et l'importance (i.e. poids) de chaque information considérée comme pertinente pour l'évaluation de la confiance. Cela nous a amené à étudier et différencier les modèles de confiance existants du point de vue des critères de confiance qu'ils utilisent.

1. Trust Management System : Système de Gestion de la Confiance

2.1 Modèles de confiance

Parmi la riche littérature sur la confiance, la vision des sociologues est particulièrement intéressante. Ils identifient deux types de facteurs qui sont largement utilisés dans l'établissement de la confiance dans les sociétés humaines : facteurs de confiance rigides (*hard*) et facteurs de confiance souples (*soft*) [14]. Les facteurs rigides représentent des informations dérivées des mécanismes de sécurité comme les *clés*, les *credentials* et les *certificats*, alors que les facteurs souples englobent les informations qui sont inférées à partir d'expériences ou d'observations (individuelles ou collectives). Sur cette base, les modèles de confiance existants peuvent être catégorisés en deux familles. Les **modèles de confiance rigides** sont dédiés à la gestion de documents signés numériquement et au respect des politiques. Ces approches présupposent, dans la plupart des cas, que la confiance est établie grâce à l'obtention de certificats satisfaisant la politique. PGP (PrettyGoodPrivacy)[1], PolicyMaker [5] et IBM TE [13] sont des exemples, parmi d'autres, pionniers de ce type d'approches. Ces modèles présentent des caractéristiques communes de la confiance : elle est directe et est accordée en fonction d'une clé d'identité [1, 5], transférable par des certificats d'autorisation [5], subjective et dépend de différentes propriétés certifiées par une autorité de confiance [13]. Les **modèles de confiance souples** soulignent principalement l'importance de la dimension sociale de la confiance [11]. Ces modèles construisent la confiance en se basant sur l'historique [15] ou les évaluations [20, 14] fournies par les autres, en l'absence ou en complément des informations personnelles.

L'intégration complète des facteurs de confiances souples et rigides n'a pas actuellement été mise en œuvre. Cependant, [6, 16] ont ouvert la voie pour de telles perspectives en explorant la combinaison des deux types de facteurs de confiance dans ce qui est appelé les *modèles de confiance hybrides*. En plus des facteurs souples et rigides, ces modèles proposent d'utiliser tout autre facteur pertinent pour l'évaluation de la confiance en s'ouvrant à toute source d'information contextuelle (e.g. le risque ou l'intérêt d'une interaction). Notre travail représente un pas de plus dans cette direction, avec comme objectif une meilleure intégration des facteurs souples et rigides dans

un modèle de confiance riche, dynamique et intégratif.

2.2 Scénario

L'innovation ouverte [9] est considérée comme une nouvelle approche pour les organisations de stimuler leur créativité et d'améliorer leurs activités R&D en mobilisant des compétences externes. Ainsi, plusieurs plateformes (e.g. Hypios.com et Innocentive.com) ont été proposées dans le but de faciliter le processus d'innovation et de bâtir des Communautés d'Innovation Ouverte (CIO). Ces plateformes constituent une vitrine pour des problématiques de recherche auxquelles les organisations sont confrontées et pour lesquelles les membres de la communauté tentent d'apporter une solution contre une rémunération. Chaque membre est libre de résoudre un problème individuellement ou de créerjoindre un groupe afin de le faire de collectivement. L'innovation au sein de ces groupes repose principalement sur la collaboration, d'où le rôle déterminant qu'occupe la confiance pour la performance collective.

2.3 Desiderata

L'étude de l'exemple ci-dessus nous a conduit à identifier trois exigences essentielles pour la gestion de la confiance dans les communautés virtuelles :

Subjectivité : Nous considérons des environnements complexes qui intègrent un large éventail d'interactions et où aucun schéma générique d'interaction ne peut être identifié. Chaque décision d'interaction nécessitant la confiance est presque unique et requiert donc des facteurs de confiance spécifiques. Un bon modèle de confiance doit être capable d'intégrer tous les critères de confiance possibles afin de permettre aux utilisateurs d'exprimer leurs exigences de confiance de manière subjective et personnalisée.

Plasticité : La confiance est extrêmement dynamique et évolue en fonction de la situation. Cela est dû, entre autre, à la versatilité des informations sur lesquelles la confiance est construite. Mais souvent, cette dynamique s'explique également par des modifications sur la manière avec laquelle ces informations sont utilisées (i.e. mise à jour du modèle de confiance). Ainsi, un modèle de confiance doit être malléable en permettant l'ajout, la modification ou la suppression des facteurs de confiance à la volée.

Influence sociale : La confiance est fortement liée aux croyances, aux valeurs, aux traditions et aux normes de la société. L'impact de l'influence sociale sur la confiance [4] n'a reçu, jusqu'à ce jour, que très peu d'attention, conduisant à des modèles complètement fermés [17]. Nous pensons que, comme toute autre pratique, la confiance est fortement influencée par le contexte social auquel nous appartenons. Dans cet article, nous nous intéressons plus particulièrement à *l'effet Asch* [4] communément appelé "conformisme sociale" [2]. On parle de conformisme lorsqu'un individu intègre ses pratiques à celle du groupe en privilégiant ces dernières en cas de conflits. Ce phénomène est d'autant plus important lorsqu'il s'agit de décisions de confiance dont les conséquences peuvent affecter l'ensemble des membres de la communauté comme c'est le cas dans notre exemple de communautés d'innovations ouvertes.

3 Fondements théoriques

Dans cette section, nous décrivons les bases théoriques et les concepts sur lesquels est construite notre proposition.

3.1 Vue générale du SC-TMS

Le modèle du système (voir fig. 1) représente un système multi-agent S défini par $S = \langle C, \mathcal{A}, \mathcal{R}, O, \mathcal{I}, \mathcal{F} \rangle$, où C est un ensemble de communautés, $\mathcal{A}$ est un ensemble d'agents, $\mathcal{R}$ est un ensemble de ressources, O est un ensemble d'opérations, $\mathcal{I}$ est un ensemble d'interactions entre agents, et $\mathcal{F}$ est une ontologie de facteurs de confiance au sein de S.

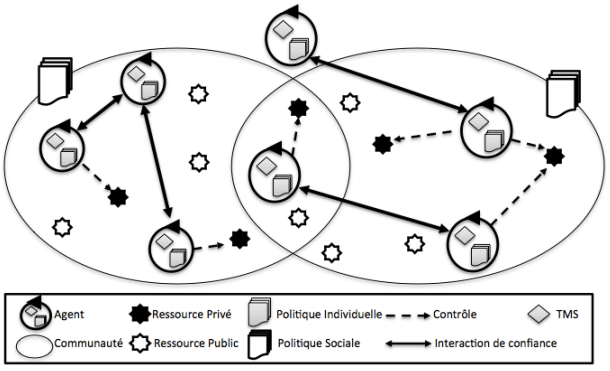

FIGURE 1 – Vue générale du SC-TMS

Définition 1 – *Les agents de S ayant des intérêts communs se regroupent pour former des*

2. On parle aussi de compatibilité sociale (Social Compliance)[10]

communautés *dans lesquelles ils interagissent fréquemment. Chaque communauté est gouvernée par un ensemble de* politiques sociales *qui décrivent les conditions minimales que les membres doivent respecter dans leur évaluation de la confiance.* $\forall c \in \mathcal{C}$*, c est représentée par* $\langle \varepsilon_c^{\mathcal{C}}, \Pi_c \rangle$ *où* $\varepsilon_c^{\mathcal{C}}$ *est l'identifiant unique de la communauté au sein de* S *et* Π_c *est un ensemble de politiques sociales imposées aux membres de c.*

Exemple 1 – *Dans notre scénario, la plateforme d'Innovation Ouverte "OpenMinds" représente le système dans lequel chaque agent représente un "expert" (i.e. fournisseur de solution). Les communautés sont constituées d'experts ayant décidé de résoudre de manière collaborative un problème donné. La communauté dont la solution est retenue en cède les droits au client contre une rémunération définie au départ.*

Définition 2 – *Les* ***agents*** *sont des entités autonomes, capables d'effectuer des opérations sur des ressources. Autonome signifie qu'aucun agent ne peut prévoir ou contrôler les opérations d'un autre agent. Chaque agent agit au nom de l'utilisateur qu'il représente et intègre un TMS chargé de prendre des décisions de confiance afin de protéger les ressources de l'agent. Un agent* $a \in \mathcal{A}$ *est défini par,* $a = \langle \varepsilon_a^{\mathcal{A}}, \kappa_a, \omega_a, \beta_a, \psi_a, \mathcal{TP}_a \rangle$*, où* $\varepsilon_a^{\mathcal{A}}$ *est l'identifiant unique de l'agent au sein de* S*,* $\kappa_a \subseteq \mathcal{O}$ *est l'ensemble des opérations que a peut effectuer (i.e.* capacités*),* $\omega_a \subseteq \mathcal{R}$ *est l'ensemble des ressources sous le* contrôle *de a,* $\beta_a \subseteq \mathcal{C}$ *sont les communautés auxquelles a* appartient*,* $\psi_a \subseteq \mathcal{R} \times \mathcal{O}$ *sont les droits que l'agent a acquis et qui établissent quelles sont les opérations qu'il peut effectuer sur chaque ressource,* $\mathcal{TP}_a$ *est un ensemble de schémas de confiance, et* $\forall tp \in \mathcal{TP}_a$*,* $tp = p_a \cup p_c \cup \mathcal{MP}_a$ *où* $p_a \in \pi_a$ *est une politique individuelle de a (et* π_a *est l'ensemble des politiques individuelles de a),* $p_c \in \Pi_c$ *est la politique sociale de la communauté à laquelle appartient a, et* $\mathcal{MP}_a$ *est un ensemble de métapolitiques de a (voir Section 4 pour une définition).*

Exemple 2 – *Alice, Bob, Carl et Dave sont quatre membres de la plateforme "OpenMinds". Ils utilisent, respectivement, les agents* $alice, bob, carl$*, et* $dave$ *pour gérer leur confiance concernant l'accès à leurs ressources critiques lors de leurs interaction au sein de la communauté. Bob, Carl et Dave souhaitent rejoindre la communauté "Com" créée par Alice pour de proposer une solution collective au problème* p_1*.*

Définition 3 – *Les* ***ressources*** *sont des artefacts passifs de* S *sur lesquels des opérations peuvent être effectuées.* $\forall r \in \mathcal{R}$*, r est représenté par* $\langle \varepsilon_r^{\mathcal{R}}, \tau_r, \varsigma_r \rangle$ *où* $\varepsilon_r^{\mathcal{R}}$ *est son unique identifiant,* τ_r *représente son type,* $\tau_r \in \{object, service, data, credential\}$ [3] *et* ς_r *qualifie sa sensibilité,* $\varsigma_r \in \{private, public\}$*. Les ressources publiques peuvent être manipulées sans restriction, alors que les ressources privées nécessitent la permission de l'agent les contrôlant.*

Exemple 3 – *Le portail d'accès aux articles scientifiques proposé par Alice représente un* service*, le cahier des charges fournit par le client, le livrable rédigé par les membres de la communauté, le fil de leurs discussions et la politique de la communauté sont des* data*, alors que les adresses électroniques et les clés PGP des membres de la communauté représentent des* credentials*. Toutes ces ressources sont* private *exceptée la politique qui est* public*.*

Définition 4 – *Les* ***opérations*** *représentent les actions que les agents peuvent effectuer sur une ressource. Sans perte de généralité, nous identifions les opérations suivantes :* $\forall o_i \in \mathcal{O}$*,* $o_i \in \tau_o = \{access, retrieve, alter, grant, release, delegate\}$ [4]*.*

Définition 5 – *Les* ***interactions*** *sont mises en œuvre par un échange de messages entre deux agents. Chaque message* $\iota \in \mathcal{I}$ *est identifiée par* $\langle x, y, \tau_\iota, o, r \rangle$*, où x et y sont respectivement l'émetteur et le destinataire du message,* τ_ι *est la force illocutoire du message avec* $\tau_\iota \in \{request, inform, permit, deny\}$ [5] *et o un ensemble d'opérations concernant la ressource* $r \in \mathcal{R}$*. Une interaction débute lorsqu'un agent* $y \in \mathcal{A}$ *(appelé le* demandeur*) envoie une requête à un agent* $x \in \mathcal{A}$ *(appelé le* contrôleur*) lui demandant la permission d'effectuer un ensemble d'opérations sur une ressource r qui est sous son contrôle.*

3. Une communauté, un groupe un rôle voire une politique peuvent également être considérés comme des ressources qu'il faut protéger.

4. Les agents on accès à d'autres opérations mais qui n'impliquent pas des ressources (e.g. join)

5. Une adaptation du standard FIPA-ACL en faisant correspondre Permit à Agree et Deny à Refuse

Définition 6 – *Une **permission**, (resp. **interdiction**) établie par un agent x pour un agent y est une déclaration établissant quelles opérations o, y à le droit (resp. n'a pas le droit) de réaliser sur une ressource r que x contrôle.*

3.2 Modèle de confiance

La figure 2 illustre le modèle opérationnel pour bâtir la confiance et représente les interactions entre les concepts requis et le processus les manipulant. Ce modèle sera mis en œuvre par chaque agent de S. Dans le modèle, chaque agent $a \in \mathcal{A}$ exprime ses exigences de confiance via des politiques de individuelles (π_a) qui sont spécifiques à chaque type de requête. Ces politiques sont dans un premier temps adaptées grâce aux méta-politiques ($\mathcal{MP}_a$) en réaction à des modifications dans le contexte, puis intégrées à des politiques sociales (Π_c) afin d'émuler le phénomène de conformité sociale décrit précédemment.

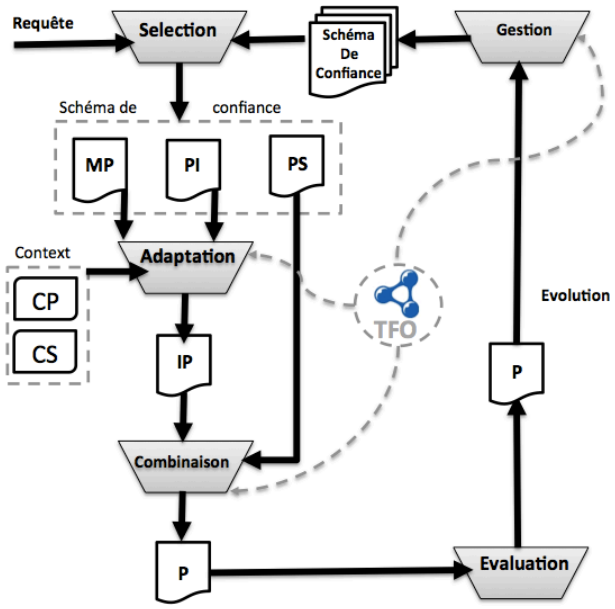

FIGURE 2 – Modèle de confiance

Les **politiques individuelles** (π_a) constituent les exigences d'un agent pour accorder sa confiance. Ces exigences sont établies en spécifiant des restrictions sur les concepts f de l'ontologie des facteurs de confiance $\mathcal{F}$ (voir section 3.3). Les **politiques sociales** (Π_c) représentent des exigences de confiance imposées par la communauté (ici c). Elles sont utilisées pour garantir un niveau de confiance minimal pour chaque interaction entreprise dans la communauté. Les **méta-politiques** ($\mathcal{MP}_a$) permettent à un utilisateur de spécifier comment ses politiques individuelles p peuvent être adaptées pour traiter certains événements. Les **schémas de confiance** ($\mathcal{TP}$) [6] représentent le moyen que nous avons choisi pour structurer et d'encapsuler les politiques et les méta-politiques en fonction du type de requête qu'elles gèrent, Toute requête q est associée à un type $\tau_q \subset (\tau_O \times \tau_R)$. Dès lors, chaque agent a est en mesure d'associer à chaque type $t \in \tau_q$ une politique individuelle p^t_a, une politique sociale p^t_c relative à la communauté à laquelle il appartient et un ensemble de méta-politiques $\mathcal{MP}^t_a$ afin d'adapter sa politique individuelle. Ainsi, seul le schéma de confiance approprié est sélectionné et utilisé, limitant ainsi les répercussions d'une erreur lors de la mise à jour d'une politique. Les informations de **contexte** sont gérées par les méta-politiques en déclenchant le processus d'adaptation des politiques. Dans notre approche, nous utilisons principalement, sans y être limité, deux sortes de contextes. Le **contexte social (CS)** a une grande influence sur les décisions de confiance. Que le contexte soit compétitif ou coopératif, différentes exigences de confiance peuvent être formulées, affectant la façon dont ces décisions sont prises. Ce modèle permet aux agents de prendre des décisions de confiance sensibles au contexte basées sur de tels indicateurs. Ainsi, des informations comme le nombre d'agents proposant la même ressource ou le nombre d'agents la désirant constituent de bons exemples d'informations de contexte social. Le **contexte personnel (CP)** sont des indicateurs que les agents utilisent pour adapter les politiques individuelles afin de correspondre aux exigences de l'utilisateur. Que la ressource demandée soit très sensible, ou que sa mise à disposition soit très enrichissante, de telles informations donnent à l'agent des informations sur les aspects sur lesquels se focaliser : protéger la ressource ou la mettre à disposition. Concrètement, dans le modèle, les politiques individuelles sont régulièrement affaiblies ou renforcées en fonction du risque ou du bénéfice pour chaque ressource et sont établies dans les préférences utilisateur.

Exemple 4 ***(Adaptation)***– *Alice estime à quatre le nombre de personnes nécessaires pour la résolution du problème p_1. Elle décide de limiter l'accès à la communauté "Com" aux candidats ayant un doctorat dans les spécialités requises. Or après quelques jours, seulement deux personnes (Bob et Carl) ont put satisfaire sa politique. Dans l'urgence et à l'approche*

6. $\mathcal{TP}$: Trust Patterns traduits en schémas de confiance

d'une date butoir, alice décide de baisser ce niveau d'exigence au master en suivant les directives que lui a laissé Alice sous forme de méta-politiques d'adaptation.
(Conformité)*– Alice autorise les membres de sa communauté à parrainer des candidats. Pour cela, une politique commune a été établie et qui impose à chaque nouveau membre parrainé d'avoir au moins un master dans la spécialité et d'avoir une réputation supérieure à "+0.5". Ainsi, afin de pouvoir décider si un membre peut être admis dans la communauté, l'agent de Bob, bob doit combiner les exigences de la communauté avec ceux requis par Bob sous peine de voir ce dernier exclus de la communauté pour non respect des exigences collectives (i.e. sociales).*

3.3 Ontologie des facteurs de confiance

L'ontologie des facteurs de confiance (voir fig. 3), ou TFO, notée $\mathcal{F}$ occupe une place centrale dans notre approche décrite dans le modèle précédent. Elle représente un référentiel commun aux agents et permet de capturer et représenter les principales exigences pour l'établissement de la confiance dans les communautés virtuelles. Selon [12], une ontologie représente la spécification d'une conceptualisation partagée. La "conceptualisation" renvoie à des choix sur la façon de décrire un domaine (e.g. exigences de confiance), la "spécification" représente sa description formelle qui est "partagée" par les membres du même système. Les ontologies sont intéressantes pour plusieurs raisons. (i) Tout d'abord, afin d'établir la confiance, un *demandeur* doit fournir des informations satisfaisant les exigences de confiance du *contrôleur*. Ces informations et leur sémantique doivent être non-ambigües et partagées par tous les agents. Aussi, (ii) la combinaison des exigences individuelles et sociales introduite par notre modèle ne peut être possible sans une description formelle et partagée de ces exigences. D'autre part, (iii) les agents qui viennent de rejoindre la communauté devront également découvrir quels sont les types d'exigences qu'ils peuvent utiliser et quelles valeurs ils doivent accepter pour leur évaluation de la confiance. Aussi, (iv) la structure représentant les exigences de confiance doit être extensible car de nouvelles exigences pourront s'avérer nécessaires. Enfin, (v) si les agents utilisant des modèles de confiances différents avaient besoin de participer à une même communauté, une étape préalable d'alignement de leurs propres ontologies le leur permettrait. Ainsi, la somme de ces besoins nous a menés à préférer les ontologies à d'autres représentations. Contrairement aux ontologies existantes [2], les concepts de la TFO sont issus de la revue de la littérature et de l'analyse des modèles de confiance existants (voir section 2).

Définition 7 *– Un* ***facteur de confiance*** $f_i \in \mathcal{F}$ *est un tuple* (τ_f, Δ_f) *où* τ_f *est le type de* f_i *et* Δ_f *son domaine (i.e. l'ensemble des valeurs que* f_i *peut prendre).*

L'ensemble des types de facteurs de confiance ($\tau_f \in \mathcal{F}$) est structuré hiérarchiquement et forme la TFO, comme illustré dans la fig. 3. Le concept racine représente le type générique des facteurs de confiance, il est divisé en deux sous-types : les *preuves* (*proofs*) qui représentent tous les facteurs de confiance *rigides*, et les *indicateurs* (*indicators*) englobent l'ensemble des facteurs de confiance *souples*. Pour simplifier, nous avons considéré dans la TFO que les *identités*, les *autorisations* et les *propriétés* certifiées sont des preuves, alors que les *expériences* (directes ou indirectes) et la *réputation* (directe ou indirecte) sont des indicateurs.

Exemple 5 *– Dans notre scénario, les agents utilisent le modèle PGP [1] pour la gestion des identités. Les autorisations représentent les permissions émises par les contrôleurs. Les membres de la communauté "Com" adoptent deux critères communs qui sont les compétences (e.g.* $\langle$"*Management*", "*Master*", 5$\rangle$*) et la réputation (e.g.* $\langle$"*Reputation*", +0.5, 3$\rangle$*).*

Comme les domaines de valeur Δ_f des facteurs de confiance sont hétérogènes (i.e. ensembles discrets et intervalles continus), et afin d'être capable d'utiliser le même mécanisme de gestion

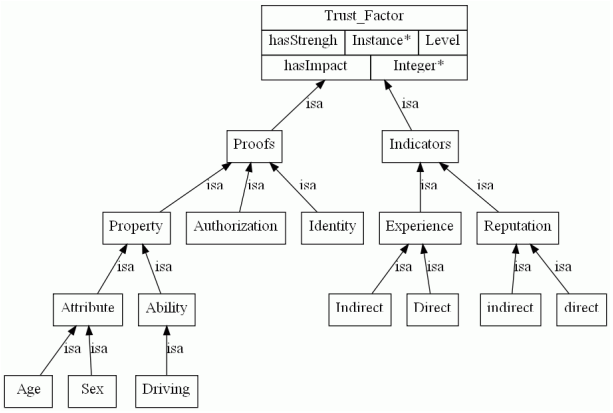

FIGURE 3 – Fragment de l'ontologie des facteurs de confiance

pour chaque $f_i \in \mathcal{F}$, nous définissons au sein de l'ontologie la relation *hasStrength* qui présente sous une forme homogène la valeur prise par chaque f_i. $\forall \tau_f, hasStrength : f_i \mapsto l \in \mathcal{L} = \{VeryLow, Low, Fair, High, VeryHigh\}$.

Exemple 6 – *Au lieu d'utiliser le facteur de confiance* $\langle$"*Reputation*", +0.5, 3$\rangle$*, les membres de "Com" pourraient utiliser* $\langle$"*Reputation*", "*Fair*", 3$\rangle$*, laissant le soin à la relation hasStrength de convertir la valeur* "*Fair*" *de manière appropriée lors de son utilisation.*

Définition 8 – *Lorsque $f \in \mathcal{F}$ est utilisé pour exprimer une exigence de confiance, nous l'appelons* ***critère de décision*** *(TC). Lorsque f est utilisé pour spécifier une information utile pour satisfaire un critère de confiance, il est appelé* ***information de confiance*** *(TI).*

Le recours à l'ontologie $\mathcal{F}$ est nécessaire dans le modèle (voir fig. 2) dans trois situations : (i) lorsque f est utilisé comme critère de confiance et que sa valeur appartient à $\mathcal{L}$; (ii) lors de l'adaptation des politiques où $\mathcal{F}$ offre alors un ordre de valeur auquel les agents se réfèrent pour renforcer ou affaiblir leurs politiques ; et (iii) lors de la combinaison des politiques individuelles et sociales où elle offre des informations permettant de choisir la politique la plus restrictive.

Exemple 7 –*Alice propose aux membres de sa communauté la possibilité de valider l'adhésion d'un nouveau membre à travers un service appelé "Adhésion". Son agent alice ne doit autoriser l'utilisation de ce service qu'aux membres ayant comme réputation* "*High*" *au sens de l'ontologie TFO. Cette exigence est définie par alice par* $TC = \langle$"*Reputation*", "*High*", 3$\rangle$*. Dès lors, pour toute requête de type* $\tau_q =$ (*access*, *service*) *à la ressource r =* "*Adhsion*"*, alice doit vérifier si carle satisfait son critère en cherchant à acquérir l'information adéquate. Par exemple, l'acquisition de* $TI = \langle$"*Reputation*", "*carl*", 0.7$\rangle$*, et en se référant aux valeurs fournies dans l'ontologie, permet à alice de constater la satisfaction des exigences spécifiées par Alice pour l'usage de se service et émettre la permission requise.*

4 Politiques de confiance

Cette section présente comment notre modèle de confiance est concrètement mis en œuvre et introduit le formalisme d'expression des politiques et les méta-politiques.

4.1 Formalisme d'expression

En plus de son expressivité , le formalisme d'expression des politiques de confiance doit être suffisamment flexible pour permettre l'adaptation et la combinaison des politiques par l'ajout, la suppression et la mise à jour des critères de confiance. Afin de répondre à ces besoins, nous avons choisi des structures de prédicats du premier ordre comme représentation de ces politiques. Prolog est ici particulièrement pertinent comme langage car il est déclaratif et peut exprimer un large ensemble de contraintes [7]. Pour les méta-politiques, Jason [8], une extension d'AgentSpeak [18] a été utilisé. Ce langage de offre des fonctionnalités permettant la combinaison des approches déclaratives et procédurales. La fonctionnalité la plus intéressante de ce langage pour la représentation de méta-politiques sont les *plans*.

4.2 Représentation des politiques

Les politiques sont représentées par un ensemble de critères de confiance. Soit $\mathcal{T} = \{tc_1, tc_2, \ldots, tc_n\}$ l'ensemble de critères de confiance. Chaque $tc_i, (i \in [1, n])$ est un triplet $\langle f_i, \aleph_i, w_i \rangle$, où $f_i \in \mathcal{F}$ correspond à un facteur de confiance existant dans l'ontologie, $\aleph_i$ représente un seuil d'acceptation (i.e. valeur ou ensemble de valeurs fournies par l'ontologie TFO), et $w_i \in \mathbb{N}^*$ représente le poids de tc_i dans la politique. Soit $p_x^t = \{tc_1, tc_2, \ldots, tc_m\}$ la politique utilisée par l'agent $x \in \mathcal{A}$ associée à la requête de type $t \in \tau_q$. La politique est assimilée à une fonction qui évalue le niveau de confiance l relatif à une requête particulière $r \in \mathcal{I}$ de type t. Alors,

$$l = p_x^t(r) = \frac{\sum_{i=1}^{m} f(\tau_i, \aleph_i, w_i)}{\sum_{i=1}^{m} w_i} \qquad (1)$$

où $f(\tau_i, \aleph_i, w_i) \in [0, w_i]$ est l'évaluation pondérée de la satisfaction de la contrainte $\aleph_i$ sur le critère de type τ_i. L'importance relative affectée à chaque critère de confiance est modélisée par le poids w_i d'un critère dans la politique. Chaque critère est alors évalué et retourne 1 ou 0, respectivement, si le critère est rempli ou non. Ce résultat est alors multiplié par le poids w_i associé à tc_i et retourné par f. Cette représentation des politiques est utilisée pour représenter à la fois les politiques individuelles et sociales.

Exemple 8 – *Soit* $r = (alter, object)$ *un type de requête. La politique individuelle que bob utilise afin d'accorder des permissions pour les requêtes de type r est définie comme*

suit : p^r_{bob} = {⟨"*Identity*","*Trustworthy*",3⟩, ⟨"*Reputation*","*Fair*",2⟩}. *Par ailleurs, la communauté Com à laquelle appartient bob oblige ses membres à respecter la politique suivante concernant les requêtes de type r :* p^r_{Com} = {⟨"*Identity*","*VeryTrustworthy*",1⟩, ⟨"*Reputation*","*High*",1⟩}. *Bob s trouve ainsi obligé d'aligner les valeurs de seuils qu'il utilise à ceux imposé par la communauté Com.*

4.3 Représenter des méta-politiques

Les méta-politiques établissent comment une politique peut être modifiée pour répondre à un contexte spécifique. Les méta-politiques présentées dans cet article sont spécifiées au moyen de plans AgentSpeak. Un plan peut être décrit comme une séquence de règles de la forme : *Upon*⟨*event*⟩ : *If*⟨*condition*⟩ ← *Do*⟨*action*⟩, où ⟨*event*⟩ représente l'événement déclencheur, ⟨*condition*⟩ est une expression générale, typiquement une conjonction ou une disjonction de littéraux à vérifier avant l'exécution du plan. Les conditions sont définies à partir d'informations provenant du contexte social, des préférences de l'agent, du *demandeur*, alors que ⟨*action*⟩ consiste en une ou plusieurs actions d'adaptation spécifiées par l'agent afin d'adapter sa politique. Le principal avantage dans l'utilisation de plans Jason est la possibilité d'exécution du code existant au travers des *actions internes* [8]. Dans ce travail, nous utilisons ces actions internes pour exécuter des opérations d'adaptation. Le résultat de l'exécution de chaque méta-politique affecte la politique initiale en ajoutant, supprimant, ou en changeant par restriction ou relâchement un ou plusieurs critères de confiance. Le processus d'adaptation est présenté plus en détail dans la section suivante.

Exemple 9 – *Alice impose à son agent d'être particulièrement vigilant quant au nombre d'adhérants dans sa communauté. Pour cela, elle a spécifié des méta-politiques afin de permettre à alice d'adapter la politique d'adhésion en fonction du contexte sociale. Soit r une requête de demande d'adhésion à la communauté, où* $\tau_r = (join, community)$ *et* p^r_a *est la politique qu'alice utilise pour prendre en charge de telles requêtes. Ces méta-politiques peuvent être définies comme suit :*

$adapt(p^r_{alice}, r, Com)$: `number(`*members*`,X) &`
`threshold(`*members*`,Y) & (X < Y)` ←
`.relax(`*reputation*`).`

$adapt(p^r_{alice}, r, Com)$: `risk(`*c*`,X) &`
`threshold(`*risk*`,Y) & (X > Y)` ←
`.restrict(`*reputation*`),`
`.add(`⟨"*identity*","*Trustworthy*",3⟩`).`

La première méta-politique réduit la réputation requise lorsque le nombre de membres est inférieur à un certain seuil alors que la seconde l'augmente et impose en plus que l'identité des utilisateurs soit établie lorsque le risque dépasse une certain niveau.

5 SC-TMS : un TMS social

Cette section présente les composants essentiels de notre TMS social, SC-TMS (ou *Socially Compliant Trust Management System*) qui assiste les agents (chaque agent étant muni d'un SC-TMS personnel) lors de leur processus de prise de décision de confiance. SC-TMS est un projet en cours de développement qui met en œuvre le modèle de confiance adaptatif présenté dans cet article.

5.1 Conception de la plateforme

L'architecture abstraite présentée en fig. 4 illustre notre implémentation. Nous utilisons la plateforme JACAMO [19] pour tirer parti des exigences discutées plus tôt, tout en préservant la généralité de l'approche. L'architecture est essentiellement composée de cinq modules responsables d'opérations spécifiques sur les politiques. Comme l'objectif principal du Framework est la gestion de politiques de confiance en réalisant les opérations présentées dans le modèle, nous présenterons le Framework par fonctionnalités plutôt que par composants. La

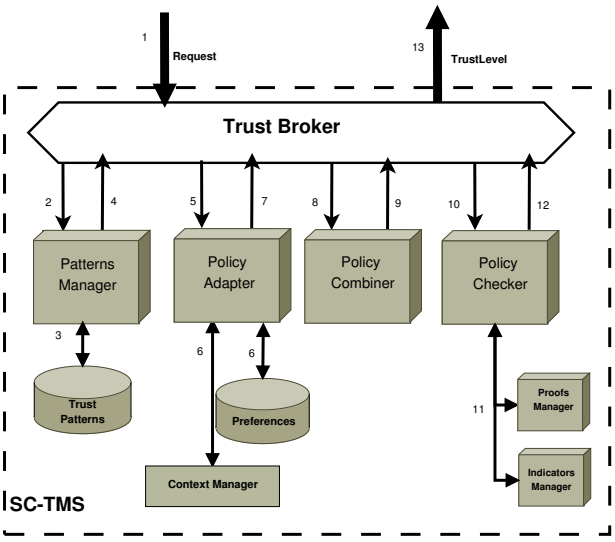

FIGURE 4 – Architecture du SC-TMS

sélection (1,2 et 4) est réalisée par le module *Trust Broker* et durant laquelle les schémas de confiance dédiés à chaque type de requête sont récupérés à partir du *Patterns Manager* en se basant sur une matrice de correspondance ($\tau_q \rightarrow \tau_O \times \tau_R$). La **gestion** (3) comporte la création et la maintenance des schémas de confiance. Elle permet à l'utilisateur de spécifier pour chaque type de requête, les politiques individuelles et les méta-politiques associées. Les politiques sociales sont automatiquement récupérées à partir de la communauté et intégrées au schéma de confiance. L'**adaptation** (5-7) est gérée par le *Policy Adapter* à travers cinq opérations dédiées : `add`, `remove`, `set`, `relax`, `restrict`. Ces opérations sont implémentés en tant qu'actions internes sous jason et leur intègration aux méta-politique permet de spécier spécifier quand et comment les politiques peuvent être adaptées. Nous illustrons brièvement dans ce qui suit comment chaque actions agit sur la politique.

- `.add(`$\langle t_i, v_i, w_1 \rangle$`)` : cette action ajoute le critère $\langle t_i, v_i, w_1 \rangle$ dans la politique courante. Si un autre TC de même type existe, une action `.set(`$\langle t_i, v_i, w_1 \rangle$`)` est déclenchée.
- `.remove(`t_i`)` : cette action supprime tous les TC de type t_i.
- `.set(`$\langle t_i, v_i, w_1 \rangle$`)` : cette action affecte les valeurs v_i, w_1, respectivement, à la force et au poids du critères dont le type est t_i. Si ce dernier n'existe pas, l'action `.add(`$\langle t_i, v_i, w_1 \rangle$`)` est enclenchée.
- `.restrict(`t_i`)` / `.relax(`t_i`)` : Ces deux actions sont assez similaires. Lorsqu'aucun paramètre n'est donné, tous les TC de la politique sont affectés par l'action. Sinon, seul le critère spécifié est affecté. Ces actions récupèrent un niveau de valeur inférieur (ou supérieur) dans la TFO pour le TC en question. Si une telle valeur existe, une opération `set` est effectuée avec la nouvelle valeur, sinon le poids du critère est diminué (resp. augmenté). Si le poids devient `null`, le TC est supprimé.

La **combinaison des politiques** (8-9) consiste à intégrer les politiques sociales (i.e. de la communauté) aux politiques individuelles (i.e. de l'agent).

Définition 9 – *Soit $\mathcal{P}_x$ l'ensemble des politiques de l'agent x. $\forall p_x \in \mathcal{P}_x$, p_x est dédiée à un type de requêtes spécifique $\tau_q \subset (\tau_O \times \tau_R)$. Soit $t \in \tau_q$, p_x^t la politique que agent x définit pour prendre en charge les requêtes de type $t = (\mu, \rho)$ au sein de la communauté c et $p_x^{\mu,\rho} \in \pi_x^{\mu,\rho}$, $p_c^{\mu,\rho} \in \Pi_c^{\mu,\rho}$ sont, respectivement, les ensembles de politiques individuelles et sociales destinées aux requêtes de type t. $p_x^t = p_x^{\mu,\rho} \oplus p_c^{\mu,\rho}$ signifie que la politique p_x^t est obtenue par **combinaison** de $p_x^{\mu,\rho}$ et $p_c^{\mu,\rho}$ de telle sorte que p_x^t soit au moins aussi restrictive que $p_x^{\mu,\rho} \cup p_c^{\mu,\rho}$*

Lorsque le *Policy Combiner* reçoit des politiques à combiner, il exécute l'algorithme suivant :

Fonction Combine($p_a^{\mu,\rho}$,$p_c^{\mu,\rho}$)

$SP \leftarrow p_c^{\mu,\rho}$; $IP \leftarrow p_a^{\mu,\rho}$; $TP \leftarrow \emptyset$; $i \leftarrow 0$
while $SP \neq \emptyset$ **do**
  foreach $SP.tc_x \in SP$ **do**
    if $tc_x.\tau_x \notin IP$ **then**
      $TP.tc_i \leftarrow tc_x$
      $SP \leftarrow SP - \{tc_x\}$
      $i \leftarrow i + 1$
while $IP \neq \emptyset$ **do**
  foreach $IP.tc_x \in IP$ **do**
    if $tc_x.\tau_x \notin SP$ **then**
      $TP.tc_i \leftarrow tc_x$
      $IP \leftarrow SP - \{tc_x\}$
      $i \leftarrow i + 1$
forall the $tc_x \in SP$, $tc_y \in IP$ *s.t.* $tc_x.\tau_x = tc_y.\tau_y$ **do**
  $TP.tc_i \leftarrow$
  $\langle \tau_i, max(tc_x.v_x, tc_x.v_y), max(tc_x.w_x, tc_y.w_y)) \rangle$
  $SP \leftarrow SP - \{tc_x\}$
  $IP \leftarrow IP - \{tc_y\}$
  $i \leftarrow i + 1$
return SP

Enfin, l'**évaluation** (10-12) est réalisée par le module *Policy Checker*. Cette partie du Framework agit comme un TMS classique dans le sens où il évalue la satisfaction de la politique en confrontant les TC aux TI reçues ou acquises. Lorsque ce module reçoit une politique à vérifier (10), et après l'avoir analysée, il identifie les types de TC. Les preuves sont demandées au *Proofs Manager*, alors que les indicateurs sont collectés par le *Indicators Manager* (11). Quand toutes les TI sont reçues, le module exécute la fonction décrite en section 4.2 afin de calculer tous les critères vérifiés. Le résultat final est une évaluation quantitative établissant le nombre de critères satisfaits parmi tous les critères formulés.

6 Conclusion et travaux futurs

Nous avons introduit dans cet article le concept de conformité sociale (i.e. respect des normes sociales) pour la gestion de la confiance au sein des communautés virtuelles. A cette fin, nous avons développé un modèle riche, dynamique et intégratif qui reproduit les caractéristiques de

la confiance (subjectivité, dynamique, sensibilité au contexte) et qui en préserve la sémantique réelle.

Nous avons d'abord exploré et isolé les facteurs de confiance dans la littérature. Puis, nous avons présenté un nouveau modèle de confiance où les exigences individuelles sont adaptées et intégrées aux exigences de la communauté. L'originalité de la proposition repose sur l'utilisation de politiques et de méta-politiques dynamiques et adaptatives afin de capturer et représenter la dynamique de la confiance. Enfin, nous avons fait les premiers pas vers le développement d'un TMS social (SC-TMS) basé sur la technologie multi-agent.

Les prochaines étapes de nos travaux incluent l'extension du modèle pour l'intégration des exigences individuelles dans les exigences sociales de manière ascendante. L'idée pourrait être d'étudier comment les agents peuvent influencer ou déclencher l'adaptation ou l'évolution des politiques sociales. Nous envisageons également d'explorer des mécanismes pour construire automatiquement des politiques sociales à partir des politiques individuelles. Une telle fonctionnalité est particulièrement intéressante pour les communautés décentralisées et auto-organisées comme les réseaux sociaux (à la *Diaspora*, par exemple).

Références

[1] A. Abdul-Rahman and S. Hailes. A distributed trust model. In *Proceedings of the 1997 workshop on New security paradigms*, NSPW '97, pages 48–60, New York, NY, USA, 1997. ACM.

[2] P. Anantharam, C.A. Henson, K. Thirunarayan, and A.P. Sheth. Trust Model for Semantic Sensor and Social Networks : A Preliminary Report. In *National Aerospace & Electronics Conference (NAECON 2010)*, 2010.

[3] D. Artz and Y. Gil. A survey of trust in computer science and the semantic web. *Web Semantics : Science, Services and Agents on the World Wide Web*, 5 :58–71, 2007.

[4] S E Asch. *Effects of group pressure upon the modification and distortion of judgments*, volume 27. Carnegie Press, 1951.

[5] M. Blaze, J. Feigenbaum, and J. Lacy. Decentralized Trust Management. In *Proc. of the 1996 IEEE Symposium on Security and Privacy*, pages 164–173. IEEE CS, 1996.

[6] P. Bonatti, C. Duma, D. Olmedilla, and N. Shahmehri. An Integration of Reputation-based and Policy-based Trust Management. In *Proc. of the Semantic Web Policy Workshop*, 2005.

[7] P. Bonatti and P. Samarati. Regulating service access and information release on the web. In *Proc. of the 7th ACM conference on Computer and communications security (CCS '00)*, pages 134–143. ACM, 2000.

[8] R.H. Bordini and J. Hübner. Semantics for the jason variant of agentspeak (plan failure and some internal actions). In *Proc. of the 19th European Conference on Artificial Intelligence (ECAI'10)*, pages 635–640. IOS Press, 2010.

[9] Henry W. Chesbrough. *Open Innovation : The New Imperative for Creating and Profiting from Technology*. Harvard Business Press, March 2003.

[10] R B Cialdini and N J Goldstein. Social influence : compliance and conformity. *Annual Review of Psychology*, 55(1974) :591–621, 2004.

[11] R. Falcone and C. Castelfranchi. *Social trust : a cognitive approach*, pages 55–90. Kluwer, 2001.

[12] T. R. Gruber. Toward principles for the design of ontologies used for knowledge sharing. *International Journal of Human-Computer Studies - Special issue : the role of formal ontology in the information technology*, 43 :907–928, 1995.

[13] A. Herzberg, Y. Mass, J. Michaeli, Y. Ravid, and D. Naor. Access control meets public key infrastructure, or : Assigning roles to strangers. In *Proceedings of the 2000 IEEE Symposium on Security and Privacy*, pages 2–14. IEEE CS, 2000.

[14] A. Jøsang. *Foundations of security analysis and design IV*, chapter Trust and reputation systems, pages 209–245. Springer-Verlag, 2007.

[15] K. Krukow, M. Nielsen, and V. Sassone. A framework for concrete reputation-systems with applications to history-based access control. In *Proceedings of the 12th ACM conference on Computer and communications security*, pages 260–269. ACM, 2005.

[16] A. J. Lee, T. Yu, and Y. Le Gall. Effective trust management through a hybrid logical and relational approach. In *Proc. of the 5th ACM Symposium on Information, Computer and Communications Security*, pages 169–179. ACM, 2010.

[17] Xin Li, Traci J. Hess, and Joseph S. Valacich. Using attitude and social influence to develop an extended trust model for information systems. *SIGMIS Database*, 37 :108–124, September 2006.

[18] A. S. Rao. Agentspeak(l) : Bdi agents speak out in a logical computable language. In *Proceedings of the 7th European workshop on Modelling autonomous agents in a multi-agent world : agents breaking away : agents breaking away*, pages 42–55, Secaucus, NJ, USA, 1996. Springer-Verlag New York, Inc.

[19] A. Ricci, J.F. Hubner, R. H Bordini, and O. Boissier. JaCaMo Project, 2010. `http://jacamo.sourceforge.net/`.

[20] J. Sabater and C. Sierra. Regret : reputation in gregarious societies. In *Proc. of the fifth international conference on Autonomous agents (AGENTS '01)*, pages 194–195. ACM, 2001.

[21] K. Seamons, M. Winslett, T. Yu, B. Smith, E. Child, J. Jacobson, H. Mills, and L. Yu. Requirements for policy languages for trust negotiation. In *Proceedings of the 3rd International Workshop on Policies for Distributed Systems and Networks (POLICY'02)*, pages 68–79, Washington, DC, USA, 2002. IEEE Computer Society.