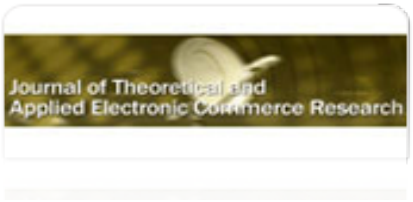

Journal of Theoretical and Applied Electronic Commerce Research
E-ISSN: 0718-1876
ncerpa@utalca.cl
Universidad de Talca
Chile

Gal-Oz, Nurit; Grinshpoun, Tal; Gudes, Ehud
Sharing Reputation Across Virtual Communities
Journal of Theoretical and Applied Electronic Commerce Research, vol. 5, núm. 2, agosto, 2010, pp. 1-25
Universidad de Talca
Curicó, Chile

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Sharing Reputation Across Virtual Communities

Nurit Gal-Oz[1], Tal Grinshpoun[2] and Ehud Gudes[3]

Ben-Gurion University, Department of Computer Science and Deutsche Telekom Laboratories
[1] galoz@cs.bgu.ac.il, [2] grinshpo@cs.bgu.ac.il, [3] ehud@cs.bgu.ac.il

Received 19 February 2010; received in revised form 10 June 2010; accepted 12 June 2010

Abstract

Trust and reputation systems for virtual communities are gaining increasing research attention. These systems track members' activities and obtain their reputation to improve the quality of member interactions and reduce the effect of fraudulent members. As virtual communities become a central playground for internet users, the reputation a member gains within a community may be viewed as a social credential. These credentials can serve the user as a means for promoting her status in new communities on one hand, and on the other hand assist virtual communities to broaden their knowledge about users with relatively low activity volume. The Cross-Community Reputation (CCR) model was designed for sharing reputation knowledge across communities. The model identifies the fundamental terms that are required for a meaningful sharing of reputation information between communities and proposes methods to make that information sharing feasible within the boundaries of users' and communities' policies. This paper presents the CR model and draws the architecture guidelines for designing an infrastructure to support it. The proposed model is evaluated by using a sample of real-world users' ratings as well as by conducting a dedicated experiment with real users. The results of the experimental evaluation demonstrate the effectiveness of the CCR model in various aspects.

Key words: Trust and Reputation Systems, Cross-Community Reputation, Virtual Communities, Infrastructure

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

1 Introduction

Some knowledge of how people in a small virtual community behave will help prevent vertigo and give you tools for comparison when we zoom out to the larger metropolitan areas of cyberspace. Some aspects of life in a small community have to be abandoned when you move to an online metropolis; the fundamentals of human nature, however, always scale up.

– Howard Rheingold, The Virtual Community, 1993

Virtual communities gather people around some common goal or shared interest. Among these one can find collaborative workgroups, illness support groups, professional consulting, intellectual discussion groups etc. These communities may be commercial or non–profitable, private or public and people may participate using their real–world identity or under disguise. Online communities and personal social networks often use reputation systems to establish trust between community members [1], [3], [7], [12]-[13]. For example, in an online marketplace a highly rated seller is more trustworthy than others with no or rather low online reputation. In a personal social network, e.g. LinkedIn (Site 1), community members are able to promote each other by writing personal comments about a colleague or business partner. This could promote getting an attractive job offer or further important contacts. As a result, reputation is a valuable asset for the community members.

The key objective of Reputation systems is to obtain and maintain measures of trust between members of a community based on their behavior during community activities. Trust and reputation models may differ in how they define the terms trust and reputation, in their assumptions on how the system obtains data on trust relationships, and in the algorithms they use to compute trust and reputation. Without being committed to specific definitions of trust and of reputation we define for the sake of this discussion that trust is a one to one relationship between two entities (e.g. users, agents) while reputation is a many to one relationship inferred by aggregation, from a group of entities that are the relying party to a single entity that is the trusted party.

Considering reputation information as part of a user's identity makes it both a sensitive and a desired data for communities to share. At the same time, a reputation that a user has gained at some point in time can leverage her state in new communities. Exchange of such reputation is a valuable resource both for the users and for the communities. *Cross-Community Reputation* (CCR) can be achieved by sharing and combining reputation data from different communities [6], [17]. The main advantages of CCR are:

- Leverage reputation data from multiple communities in order to produce more accurate recommendations. This is especially important when users are active in several communities and may receive different valuations in those communities.
- Accumulate Reputation – A user does not necessarily have to build reputation from scratch when joining a new community.
- Enable users to maintain (either global or community-specific) offline reputation certificates. This is known as *reputation capital* [15].
- Boost new virtual communities by importing reputation data from related communities.
- Facilitate business cooperation between communities of similar domain.

The task of computing CCR is far from trivial. Each community has its own way of perceiving reputation and computing it and may include different factors in its reputation mechanism. In order to compare between different mechanisms we refer to the following basic properties of trust relations as proposed in [14]: *Measure* – the value domain of the reputation score; *Trust context* – the context to which a reputation score is assigned; *Certainty* – a measure expressing the level of confidence a community has in the firmness of the reputation level. These properties are the foundations of our model. Dealing with the conceptual complexity of CCR is a major goal of the present paper.

The issue of transferring reputation data between agents was studied by several researchers. The use of a common ontology in order to exchange reputation between agents is proposed in [18]. Several preliminary ideas for translating recommendations are proposed in [4]. Realizing that reputation is a valuable resource for both the users and the communities lead several researchers to study the idea of cross-community reputation [6], [17].

A number of papers on Trust and Reputation discuss aspects important for cross-community reputation. The problem of uncertainty and confidence in computing reputation is discussed in [12], [20]. The Beta reputation model [12] presents a formal framework for computing the uncertainty in a single community case. The problem of providing privacy while transferring reputation from one community to another was investigated in [17]. Lee and Yu [16] introduce the idea of a composite model of trust that allows the composition of both reputation data (horizontal trust)

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

and credentials (vertical trust). Their paper suggests a new policy language that allows such compositions. However, it does not deal with the computation of the aggregated trust. Some commercial products provide a very simple notion of cross-community reputation, e.g. iKarma (Site 2) and TrustPlus (Site 3), which do not exhibit the complexities of the concept as presented here.

The present paper introduces a model for computing CCR (see also [8]). A community that wishes to receive CCR data regarding one of its users sends a request to relevant communities either directly or through a trusted third party. Communities that have reputation data of the user and are willing to share the information reply with the relevant reputation data. The received data is assembled into an object containing the CCR data of the user in the context of the requesting community.

Reputation is a complex term that cannot always be quantified into a single value, but is rather composed of several attributes. In order to provide a common and agreeable notion of reputation we propose to define a set of generic attributes. A virtual community should follow this set of attributes, or at least define a mapping from its own internal attributes to the generic set. For instance, a generic set for virtual marketplaces may include the following attributes – *Reliability*, *Promptness*, *Customer Service*, *Warranty*, and *Price*. Existing communities may already have their own attributes for reputation. For instance, eBay (Site 4) has four attribute criteria by which a seller is rated – *Item as Described*, *Communication*, *Shipping Time*, and *Shipping and Handling Charges*. In order to enable CCR with eBay, a mapping between eBay's criteria and the set of generic attributes must be added. Attribute matching is a general problem that appears in many domains such as Semantic Integration, Schema Matching (see e.g. [19]) and Ontology based Web Search [24]. However, its application to cross-community reputation as described in the model presented here is new.

In addition to the CCR model this paper proposes an infrastructure for computing and exchanging *Trust and Reputation In virtual Communities* (TRIC). TRIC stands for an implementation project of the proposed architecture (see also [5]). In its core TRIC consists of two fundamental computational models – one for the internal reputation within each community and one for the sharing of reputation knowledge across communities. TRIC can use any Trust and Reputation (T&R) model for the computation of the internal reputation within each of the communities (e.g. [1], [3], [12]-[13]). In the current version of TRIC we assume the use of the Knot model [7] for computing the internal reputation, but this does not limit the generality of the architecture.

TRIC uses the CCR model (section 3) for the computation of shared reputation knowledge across different communities. The CCR model supports the transfer and matching of reputation from heterogeneous sources that use different computation and scaling mechanisms for obtaining reputation.

Internet users may join several communities acting around their area of interest or expertise. Community users may use their real-world identity or some virtual identity provided by an identity provider, e.g. MyOpenID (Site 5). In some cases these users keep their identity hidden, while in other cases they use the same identity for several communities. Usually communities have no information of their users beyond the context of the community. Interesting information with this respect may be a user's true identity, the other communities she is active in, etc. Consequently, an important issue in TRIC is the users identity management, especially in supporting their anonymity and privacy.

The major contributions of the TRIC framework for enabling CCR are:

- Presenting a general and modular architecture for supporting Trust and Reputation in and across multiple communities. The architecture can support any T&R mechanism and a most general CCR model.
- Supporting anonymity and privacy of users as a main goal of the architectural design.
- Supporting standard identity management and authorization services, thus enabling the migration of the technology to real products.
- Identifying the design issues required for a highly reliable and scalable system.

In addition to presenting the CCR model and the TRIC infrastructure, an important contribution of this paper is a detailed experimental evaluation that demonstrates the effectiveness of CCR using real data.

The rest of the paper is organized as follows. Section 2 provides an overview of the TRIC project. Section 3 presents a detailed description of the CCR model and is followed by a step-by-step example of the CCR computation process (section 4). The TRIC framework along with its implementation issues are detailed in sections 5 and 6, respectively. The experimental evaluation of the CCR model is given in section 7 and section 8 concludes the paper.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

2 The Scope of TRIC

When discussing TRIC in the context of virtual communities we consider four layers: end-user client (community member), application layer (community), TRIC client, and TRIC Server. These layers are illustrated in Figure 1.

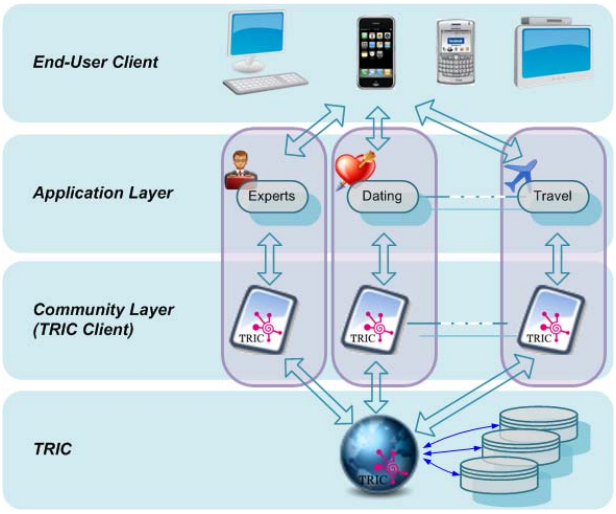

Figure 1: TRIC in the Context of Virtual Communities

- **End-user client** refers to the software application, which enables a user to display and interact with text, images, videos, music and other information located on the application layer, e.g. a web-browser based client.

- **Application layer** refers to the different applications (communities), where each application exposes its own graphic web pages, and its specific (and probably different from others') functionality.

- **Community layer (TRIC Client)** provides the application with the necessary functionality required to manage a community. Specifically in the scope of the TRIC project, this module includes the interfaces to TRIC Services (T&R Services and CCR services) and is embedded in the communities' application logic.

- **TRIC Server** is the infrastructure that provides the communities with T&R and CCR services.

A TRIC operator manages an instance of the TRIC infrastructure and may provide CCR services to orbiting communities. The TRIC operator provides the registered communities with a TRIC client for both the CCR services and the T&R services (for computing internal reputation within the community). Once a community is registered to TRIC it uses a TRIC client to interface with the TRIC server. A Community should have the ability to determine the reputation of its users within the community. It may have its own T&R system for doing so or it can use the T&R services offered by TRIC.

A very important motivation for the current design of TRIC is the support of *unlinkability* (see section 6.1). One may consider a completely distributed architecture in which communities communicate directly with other communities; however such architecture cannot support unlinkability, since the common identification must be known to all participating communities. In addition no central identity management by a trusted provider which may be desired by users can be supported, and of-course no centralized monitoring which can be helpful for detecting misbehaving users can be supported.

The CCR computation process [8] begins when a *requesting community* that wishes to receive CCR data regarding one of its users, sends a request to relevant *responding communities* (either directly or through a trusted third party, TRIC in our case). Communities that have reputation data of the user and are willing to share the information, reply with the relevant reputation data. The received data is assembled into an object containing the CCR data of the user in the context of the requesting community. This process is illustrated in Figure 2: (1): A requesting community sends TRIC a request for the CCR of a community member; (2): TRIC compiles a request and (3) submits it to all potential responding communities; (4): Responding communities submit a reputation object of the member at subject; (5): TRIC processes all reputation objects and compiles a CCR object; (6): TRIC sends the CCR object to the requesting community.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

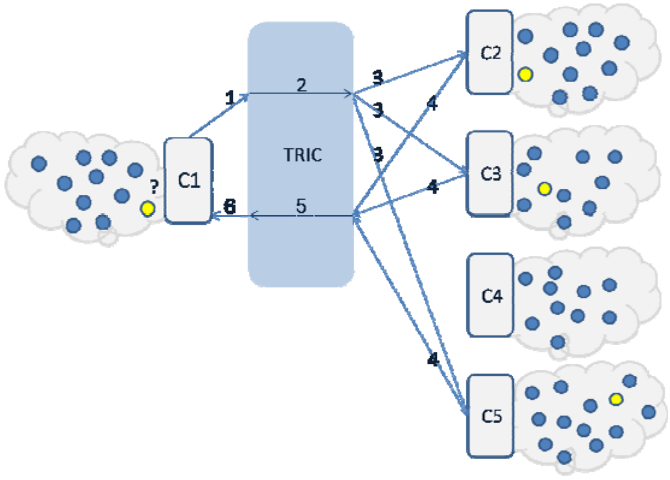

Figure 2: Request for CCR scenario

3 The CCR Model

The assumed scenario is of a user active in several communities that maintain her internal reputation. The communities are willing to share reputation data as well as some other community related meta-features in order to enable a precise and reliable computation of CCR.

The CCR Model consists of three major stages – preconditions, conversion of reputation values and attribute mapping. These stages must be performed for each pair of communities that wish to enable CCR between them. While the first stage is performed only rarely, the stages of conversion of reputation and attribute mapping take place in every request for CCR. Additionally, reputation data privacy issues are controlled through policies defined by both the user whose CCR we seek, and the community that applies for her CCR. We define the major actors and entities in our model as follows:

CCR service – a service provided by a third party to enable sharing of reputation information between communities.
Communities – a community $\in$ *CCRcom* is a community registered for the CCR service that uses some trust and reputation model to evaluate its users' reputation internally.
User – a user $\in$ *Users* is a participant in one or more communities in *CCRcom*, registered for the CCR service.
CCR Request – A request issued by a community for the CCR value of one of its users.
Requesting Community – A community $\in$ *CCRcom* that initiates a request for the CCR of one of its users.
Responding Community – A community $\in$ *CCRcom* capable of providing a reputation value for any of its users, in response for a CCR request.

Any community may act as a requesting community in some CCR requests, and as a responding community in others.

3.1 Enabling Preconditions

The cross-community reputation model enables communities to anonymously participate and share information with other communities. Inherent with the benefits of having a wider reputation base, there is a risk of getting inadequate information from communities that are not compatible or relevant for some reason. To minimize this risk we maintain confidence values that express the level of trust each community has in another. The confidence value is a unique value inferred from the nature of the two communities and statistical information collected on past CCR computations.

Definition 1 *Confidence*(*A*,*B*) is a number in the range of [0..1] representing the extent to which a requesting community *A* considers the input from a responding community *B* as relevant and valuable for CCR computation.

A confidence level lower than a predefined confidence threshold implies that the community is unreliable and cannot serve as a responding community. Any other value of confidence is used as a community's weight when combining reputation scores from different communities, as will be shown in sections 3.3 and 3.4.

The confidence one community has in another is determined by three factors: *Explicit Assertion, Category Matching level* and *Domain Confidence*. While category matching level and domain confidence are objective measures used to compare two communities, an explicit assertion is a subjective statement of confidence the owners of one

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

community have in another. Following is a formal definition of these factors and the computation of confidence level. For convenience, we use normalized values in the range [0..1] to represent confidence.

Explicit Assertion. Each community may explicitly express its subjective confidence level in other communities.

Definition 2 *Explicit Assertion* is a *confidence* value explicitly provided by a representative of a community with respect to another community.

Explicit assertions may be viewed as a mean to create ”black lists” of communities perceived as incompatible on any grounds (e.g. ethical, political) and ”white lists” of communities that are considered valuable (e.g. professional, popular). Therefore, explicit assertions are either used to block some reputation inputs or to enforce the use of other reputation inputs regardless of any computational result. Explicit assertions can be edited either manually by an administrator or automatically according to statistical information pointing out the significance of one community as a source of reputation for another community.

For each community $A \in CCRcom$ there is a set of assertions (possibly an empty set) denoted by *ExpAssertions*(*A*) of explicit confidence statements that may be provided regarding any other community $B_i \in CCRcom$:

(1)
$$ExpAssertions(A) = \{(ExpConf(A,B_i))|ExpConf(A,B_i) \in [0,1]; A,B_i \in CCRcom; A \neq B_i\}$$

Being the subject of a CCR request, a user can also define a set of explicit assertions with respect to some communities. In this case the two sets are unified.

Community Category Matching. A community may belong to one or more categories representing aspects of its activity. Communities belonging to the same category are more likely to rely on reputations they exchange, assuming the motivation and user ranking criteria they use are similar. A simple approach for community categories matching is to characterize them by keywords. The characterization of a community consists of the set of keywords that correspond best with the activities of the community. These keywords may be taken from the community’s web pages meta-tags (routinely used for promoting placement in search engines).

Definition 3 *Category Matching level* is a value in [0..1] representing the correlation of two communities based on their categories as described by keywords.

One of the association measures commonly used in information retrieval for keywords based matching is the Dice Correlation Coefficient. Let *CM* be the category matching function, *KW(A); KW(B)* two sets of keywords representing communities *A* and *B*, then using Dice coefficient:

(2)
$$CM(A,B) = \frac{2|KW(A) \cap KW(B)|}{|KW(A)| + |KW(B)|}$$

Other approaches for automatic community categorization and category matching level detection include algorithms for document matching based on the vector space model [21]. In these algorithms, documents are represented as vectors where each element is associated with a word in a predefined vocabulary. In the proposed model communities’ web pages may provide the content of these documents (e.g. the ”About” page). Each word in the vocabulary of a community vector is assigned a weight representing its importance to the community, and considering its importance within all communities in *CCRcom* (e.g., using the TF-IDF measure [11]).

Domain Confidence. Communities use various reputation models that may each utilize a different representation of reputation values. Cross-community reputation exchange may require conversion from one domain of values to another. However, some domains are more granular than others and provide a more precise representation that may be compromised when computed by conversion from a less precise representation. In this case the confidence in the accuracy of the converted value decreases.

Pinyol *et al.* [18] address this problem as part of the reputation ontology and mapping mechanism that they introduce. We adopt a relaxed version of their approach and focus on the following four domains of reputation representation: Boolean (*BO*), a discrete set of 5 values (*DS5*), a discrete set of 10 values (*DS10*), real numbers bounded by a predefined range of [0..1] (*RE*).

All four domains are considered as discrete sets while real numbers are represented using 100 possible values. They assign an uncertainty value to every possible conversion from one domain to another, reflecting the possible loss of data caused by the conversion. The computation is based on the Shannon entropy [23] of each of the considered domains. The entropy of conversion is computed by summing up all conditional entropies of a variable in the target domain given every possible value in the source domain (for details see [18]). Using this method, a domain conversion can be carried out for any two domains of values. The four domains presented here are an example.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Table 1(a) is a generic table presenting the uncertainty resulting from conversion of reputation values from one domain to another, based on the entropies of the source and target domains. The table shows that although there is no information loss when converting from a more general domain (e.g. Boolean) to a more expressive domain (e.g. Real), the conversion injects some uncertainty to the values on the target domain. This results from the fact that there may be more than a single value in the target domain that could have been selected to represent a single value in the source domain.

Table 1: (a) Conversion uncertainty, (b) Domain confidence

(a)

	BO	DSS	DS10	RE
BO	0	1.29	2.32	5.64
DSS	0	0	1	4.32
DS10	0	0	0	3.32
RE	0	0	0	0

(b)

	BO	DSS	DS10	RE
BO	1	0.89	0.79	0.50
DSS	1	1	0.91	0.62
DS10	1	1	1	0.70
RE	1	1	1	1

Clearly, a community should not ignore inputs from another community based solely on reputation representation differences. On the other hand, it should be able to express the fact that these inputs are approximated.

Definition 4 *Domain Confidence* is a value in [0..1] representing the extent to which one community considers the input from another community as precise, based on conversion uncertainty.

Based on conversion uncertainty we define the maximum confidence *MaxC* we have when accepting an input from the most general domain (*BO*) to the most expressive domain (*RE*), i.e., when the potential loss of information is the highest. Formally, we infer the domain confidence (*DC*) of community A in community B based on the conversion uncertainty *CU(A,B)* as follows:

(3)

$$DC(A,B) = 1 - \frac{CU(A,B)}{CU(BO,RE)} \cdot (1 - MaxC)$$

Table 1(b) summarizes the domain confidence values using *MaxC* = 0.5.

Computing Confidence Level. Explicit assertions are subjective statements that may or may not agree with the information inferred by comparing community categories or other aspects of reputation mechanisms. It may be motivated by the will to be influenced by certain communities considered prestigious or professional, as one may wish her own community to be. In our model, explicit assertions override any other information regarding the confidence of one community in another. In the absence of an explicit assertion we consider both domain confidence and category matching level when calculating the confidence of a requesting community in a responding community. We use the confidence value first to filter out communities previously identified as unreliable sources for reputation. For responding communities that passed that filter, we use the confidence of the requesting community in the responding community as a weight factor, reflecting the relative importance of the inputs provided by that community.

(4)

$$Confidence(A,B) = \begin{cases} ExpConf(A,B) & ExpConf(A,B) \in ExpAssertions(A), \\ CM(A,B) \cdot DC(A,B) & otherwise \end{cases}$$

3.2 Conversion of Reputation Values

Due to differences in reputation representation in different communities, reputation values from responding communities must be converted to values in the requesting community's domain before they can be used. The present paper adopts the mapping approach proposed by Pinyol *et al.* [18] to carry out this conversion, and provide a simplified version of this approach. Mapping from one discrete set domain to another involves partitioning the more specific set to subsets. The number of subsets is determined by the cardinality of the less specific set. The mapping is then straightforward from a value that belongs to the i^{th} subset in one domain to the i^{th} value of the second domain and vice versa (the median of each subset may represent it for this matter).

The mapping is formalized as follows: Assume that a discrete set of n values DS_n, is an ordered set of n values labeled as 0,…,n-1. Let DS_m, DS_n be two discrete sets of size m, n respectively $m<n$. Let $DSS_n(i)$, i = 0..m−1 be m disjoint ordered sub sets of the set DS_n such that for any two items labeled as k, l, where $k \in DSS_n(i)$, $l \in DSS_n(j)$; $k < l \Rightarrow i < j$.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

If $v \in DSS_n(i)$ and $minLabel(DSS_n(i)), maxLabel(DSS_n(i))$ are the smallest and largest labels in $DSS_n(i)$ then:

(5)
$$\begin{cases} domainMap(v,n,m) = i, \\ domainMap(i,m,n) = median(minLabel(DSS_n(i)), maxLabel(DSS_n(i))) \end{cases}$$

The cardinality of each subset is not necessarily the same and may be subject to additional information about the domain. For example, a discrete set of 5 values {*A*,*B*,*C*,*D*,*F*} reflecting the ECTS grading system may be mapped to a discrete set of 100 values using the following partition: {90-100},{80-89},{70-79},{60-69},{1-59}.

Statistical Adjustment. A reputation score can be more meaningful when one knows the distribution of ratings in the community it originated from. The same reputation score can be perceived as exceptionally high or as an average popular score, under different distributions. The standard score of a variable in a given distribution indicates its location (distance from the mean) within this distribution in units of standard deviation. Adjusting a reputation value provided by a responding community to preserve its standard score ensures that the value will be equivalently positioned within the distribution of the requesting community. This adjustment is done under the assumption of normal distribution in both communities after a cross domain conversion.

Let v be a reputation value provided by a responding community B, $Z(v)$ be the standard score of v, and σA, μA the standard deviation and mean of the ratings in community A. v is adjusted in community A as follows:

(6)
$$StatAdj(v,A) = (Z(v) \cdot \sigma_A) + \mu_A$$

3.3 Attribute Mapping and Computation

Reputation is usually represented by more than a single value. The rating criteria used within a community to evaluate a transaction serve as the set of attributes describing the reputation. An attribute in one community may have the same meaning as an attribute in another community even if they are labeled differently. On the other hand, an attribute may be only partially (or not at all) analogous to one or more attributes used by a different community.

To obtain the relative contribution of an attribute of one community to the CCR computation of another community's attribute, a set of generic attributes is defined. Generic attributes correspond to the rating criteria commonly used by the participating communities. Each community provides a mapping of its attributes to the relevant generic attributes. This mapping specifies the generic attributes that match each of the community's attributes and the level of matching. This information along with the actual attribute scores provided by the responding communities enables the computation of the CCR attribute scores and the level of certainty one has in the firmness of each of these scores.

Definition 5 *Matching Level* is a number in the range [0..1] specifying the extent to which the meaning of one attribute is considered analogous to that of another attribute.

Let $GenericAtt = \{GAtt_1,\ldots,GAtt_n\}$ be the set of generic attributes, and let $\{Att(A)_1,\ldots,Att(A)_s\}$ be the set of attributes used in a requesting community A. For each attribute $Att(A)_i$, i=1..s there is a mapping to each attribute in $GenericAtt$ that it matches denoted by $Att(A)_i.ML(GAtt_j)$.

The process of computing the score and the certainty of an attribute of the requesting community has two parts. In the first, the score and the certainty of each of the relevant generic attributes are evaluated from their matching attributes in the responding communities. In the second part, the score and the certainty of the attribute at subject are evaluated from the generic attributes' scores and certainties as computed in the first part.

Let $Att(B_i)=\{Att(B_i)_1,..,Att(B_i)_{s_i}\}$ be the sets of attributes of $B_i \in Bres$, the set of responding communities that passed the confidence threshold for A. Following that, $Att(B_i)_l.Score$ is the score of the inquired subject for attribute l in the responding community B_i, and $Att(B_i)_l.Support$ corresponds to the number of ratings that constitute the score. The certainty and score of the generic attribute $GAtt_j \in GenericAtt$ with respect to Community A is given by:

(7)
$$GAtt_j(A).Certainty = \sum_{B_i \in B_{res}} \sum_{l=1}^{s_i} Att(B_i)_l.ML(GAtt_j) \cdot Confidence(A,B_i) \cdot Att(B_i)_l.Support$$

(8)
$$GAtt_j(A).Score = \frac{\sum_{B_i \in B_{res}} \sum_{l=1}^{s_i} Att(B_i)_l.Score \cdot Att(B_i)_l.ML(GAtt_j) \cdot Confidence(A,B_i) \cdot Att(B_i)_l.Support}{GAtt_j(A).Certainty}$$

To compute the score and certainty of an attribute $Att(A)_k$ for the requesting community, we use only the set of generic attributes that match it with matching level > 0 denoted by $GenAtt(Att(A)_k)$.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

(9)

$$CCR.Att(A)_k.Certainty = \sum_{GAtt_j \in GenAtt(Att(A)_k)} Att(A)_k.ML(GAtt_j) \cdot GAtt_j(A).Certainty$$

(10)

$$CCR.Att(A)_k.Score = \frac{\sum_{GAtt_j \in GenAtt(Att(A)_k)} GAtt_j.Score \cdot Att(A)_k.ML(GAtt_j) \cdot GAtt_j(A).Certainty}{CCR.Att(A)_k.Certainty}$$

We assume that the domain of values of all attributes within a community is the same and aligned with the domain of the aggregated reputation value. The conversion stage described in section 3.2 must be applied to every attribute score provided by a responding community before the above computation is carried out.

3.4 Compiling CCR

A *reputation object* provided by a community contains in addition to the single reputation score, information related to the attributes of that community (score and certainty) and statistical information related to the reputation value (e.g. its standard score). A reputation object may also include textual comments written by users who have rated the user that is the subject of the reputation object.

A CCR object is a reputation object containing the aggregated attributes values (score and certainty) that are computed from all responding communities. It also contains a CCR single score computed from the CCR attribute values using the attributes-weights assigned by the requesting community.

Let $w_1,..,w_n$ be the weights that the requesting community *A* assigns to its reputation attributes $Att_1,..,Att_n$

(11)

$$CCR.SingleScore = \frac{\sum_{i=1}^{n} CCR.Att(A)_i.score \cdot w_i}{\sum_{i=1}^{n} w_i}$$

The weights used for each CCR attribute in the CCR single score computation are defined by a community. Clearly, in communities that support user defined weights to calculate internal reputation scores, the CCR single score can be computed by the community, customizing the weights to the preferences of the viewing user (see [7]).

In addition we compute an inscrutable reputation that ignores attribute information. It is calculated as a weighted average of all single reputation scores provided by the responding communities, weighted by the confidence that the requesting community has in them:

(12)

$$CCR.InscrutableReputation = \frac{\sum_{B_i \in B_{res}} Confidence(A, B_i) \cdot B_i.ReputationScore}{\sum_{B_i \in B_{res}} Confidence(A, B_i)}$$

where $B_i.ReputationScore$ is the reputation value provided by community B_i.

This score is important when there are relatively few attributes in the responding community that match the attributes of the requesting community. Unlike the CCR single score, it is insensitive to the internal representation of the responding communities' reputation scores and it does not express the weights of reputation attributes.

3.5 CCR Policies

In addition to the actual computation of CCR, communities and users are able to define their own policies with respect to the level and type of information they are willing to reveal on top of a single aggregated reputation score. Each community may define a policy in which it specifies whether it allows as a responding community, sharing of reputation attributes, the identity of the community, additional textual comments that users attach to ratings, etc. Similarly, each user may define her own policy in any community she belongs to, to be used when the community takes the role of a responding community. Specifically users and communities (and TRIC administrator) may define disclosure policies for the following:

- Explicit assertions – whether to accept reputation at all from a given community
- Source community (and reputation value) – whether to disclose the source (and local reputation value) of each responding community contributing to the CCR computation
- Reputation details – weights of attributes, date computed, textual comments, etc.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

These policies are unified to reflect the most restrictive policy and are used when compiling the final CCR Object. Privacy restrictions may affect the level of drill down available from the compiled CCR to the input provided by each of the responding communities. The CCR object returned to the requesting community will consist of at least a single reputation value. All other information may be restricted by the unified policies. TRIC enables an interface by which the above policies can be specified and the TRIC component responsible for enforcing policies is the *Policy Agent* described in section 5.2.

Choosing between *open* and *closed* policies can impact the reliability of the CCR value. For example in many hotel websites, users may see explicit ratings and textual comments given by other users. This increases the trust that is assumed for such ratings (see [22] for the tradeoff between privacy and trust).

Attacks Mitigation Policies. The quality of reputation systems is measured mainly based on its ability to provide accurate reputation values. The task becomes much harder to achieve under the assumption of possibly dishonest participants. The major concern is that members of a community will attempt to manipulate the system to their own benefit. Hoffman *et al.* [10] investigate the subject of attacks against reputation systems, and survey existing defense mechanisms. The CCR model relies heavily on the robustness of these reputation systems. Within the TRIC infrastructure we propose a reputation system (T&RP) based on the Knot model [7] that addresses the problem of colluding members and selfish peers (within a single community). However since each community may have its own reputation system which handles attacks in different ways, the problem may be amplified when CCR is at subject. Consider for example a traitors attack [10], in which colluding users succeed to subvert the results of one or more members in their community. The reputation of these members is transferred as an input for CCR requests. In case the attacked community is the only responding community in this scenario, the CCR results will be directly affected. However, one of the purposes of having TRIC as a trusted third party is accountability. By mining results of CCR requests and responses anonymously over time, we can identify a community that provides significantly different results. TRIC can notify the community of a possible attack. Following that, the community is expected to take action (e.g. monitoring or blocking of suspected members). In addition, in order to assure the reliability of the CCR computation, TRIC may use this information to decrease that community's weight as a responding community in the absence of an explicit assertion from the requesting communities. The penalty when using such a policy is inflicted on the community for not providing a resilient reputation system. This may be perceived as decreasing the community's reputation as a reputation-provider. The problem at the user level remains impractical to solve unless real world credentials are required for registering to TRIC. If only virtual identity is required, there is always a risk for attacks under multiple identities.

4 A Detailed Example

Our example is based on three well known communities that deal with hotel recommendations – TripAdvisor (Site 6), Expedia (Site 7), and Booking.com (Site 8). Following is a description of the generic attributes and the communities:

- *Generic Attributes* – We use a set of 6 generic attributes common for the hotel recommendations domain: Comfort, Clean, Maintenance, Staff, Extra Services (*ESer*), and Value.

- *TripAdvisor* – Rooms, Service, Value, Cleanliness, and Dining are the attributes by which a hotel is rated in TripAdvisor (*TA*). In order to enable CCR with TripAdvisor, a mapping from TripAdvisor's attributes to the generic attributes should be added. Some attributes can differ solely by their names making the mapping trivial, e.g., $\text{Clean}_{(Generic)} \leftarrow \text{Cleanliness}_{(TA)}$. Other attributes might have a partial correlation between them, e.g., $\text{Comfort}_{(Generic)} \leftarrow 0.9 \cdot \text{Rooms}_{(TA)}$, $\text{Maintenance}_{(Generic)} \leftarrow 0.2 \cdot \text{Rooms}_{(TA)}$. Extra Services can be considered as a combination of Dining and Service ($\text{ESer}_{(Generic)} \leftarrow 0.7 \cdot \text{Dining}_{(TA)} + 0.3 \cdot \text{Service}_{(TA)}$). TA presents the reputation scores as stars (including half-stars), so it uses a *DS10* value domain.

- *Expedia* – The attributes used to rank the hotels in Expedia are Hotel Service (*HSer*), Hotel Condition (*HCon*), Room Cleanliness (*RCle*), and Room Comfort (*RCom*). Notice that some generic attributes may have no mapping at all from a particular community (Value in the case of Expedia). Expedia presents the reputation scores as real numbers (using one decimal place precision) ranging from 0 to 5, so it uses a *DS50* value domain.

- *Booking.com* – The attributes used to rank the hotels in Booking.com are Staff, Services, Clean, Comfort, and Value for Money (*VFM*). Booking.com presents the reputation scores as real numbers ranging from 0 to 10, so it uses a *RE* value domain.

Let us first determine the level of confidence between each pair of communities. For clarity we simplify our example by making several assumptions. First, we assume that in all three communities, a rating of zero does not exist. Second we assume full category matching and similar reputation mechanisms except for the value domains. These assumptions are reasonable, since all of the discussed communities are websites that deal with hotel recommendations. We also assume the absence of explicit assertions.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Table 2 presents the domain confidence as derived from the values in Table 1(b). Due to the assumptions, the values in Table 2 determine the level of confidence between the communities.

Table 2: Domain confidence in the example

	TripAdvisor	Expedia	Booking.com
TripAdvisor	-	0.79	0.71
Expedia	1	-	0.91
Booking.com	1	1	-

Consider three hotels in Milan, Italy – Enterprise Hotel (*H*1), Brunelleschi Hotel (*H*2), and Ripamontidue Hotel (*H*3). H1 and H2 are popular hotels that appear in all 3 communities, while H3 can only be found in TripAdvisor and Expedia. Table 3 shows the reputation scores of the hotels in each community, while Table 4 summarizes the attribute mappings of all the communities in the example.

Table 3: Reputation scores in: (a) TripAdvisor, (b) Expedia, (c) Booking.com

(a)

	H1	H2	H3
Rooms	4.0	3.5	3.5
Service	4.5	3.5	3.5
Value	4.5	3.5	3.5
Cleanliness	4.5	4,0	3.5
Dining	3.5	4.0	3.0
Support	157	284	151

(b)

	H1	H2	H3
HSer	4.6	4.2	3.5
HCon	4.7	4.0	3.8
HCle	4.7	4.4	4.2
RCom	4.5	4.0	3.5
Support	19	68	4

(c)

	H1	H2
Staff	8.5	8.2
Services	8.8	8.0
Clean	9.1	8.6
Comfort	8.8	8.1
VFM	8.3	7.6
Support	240	428

Consider three hotels in Milan, Italy – Enterprise Hotel (*H1*), Brunelleschi Hotel (*H2*), and Ripamontidue Hotel (*H3*). H1 and H2 are popular hotels that appear in all three communities, while H3 can only be found in TripAdvisor and Expedia. Table 3 shows the reputation scores of the hotels in each community including their support. In these communities the same support applies to all the attributes. Table 4 summarizes the attribute mappings of all the communities in the example:

Table 4: Attribute mapping in the example

Generic	TripAdvisor	Expedia	Booking.com
Att1 (Comfort)	Rooms (0.9)	RCom	Comfort
Att2 (Clean)	Cleanliness	RCle	Clean
Att3 (Maintenance)	Rooms (0.2)	HCom (0.8)	
Att4 (Staff)	Service (0.9)	HSer (0.5)	Staff
Att5 (ESer)	Dining (0.7) Service (0.3)	HSer (0.4)	Services
Att6 (Value)	Value		VFM

Consider Expedia that requests CCR for the Enterprise Hotel. There are two communities that can respond to this request – TripAdvisor and Booking.com. In its request, Expedia needs data only for the generic attributes Att1–Att5, since it has no internal attribute that is mapped to the generic attribute Att6 (see Table 4).

The computation of the CCR certainty for the generic attribute Att1 (*Comfort*) is as follows:

$$Att1.Certainty = Rooms_{(TA)}.ML \cdot TA.Conf \cdot TA.Supp + Comfort_{(Booking)}.ML \cdot Booking.Conf \cdot Booking.Supp$$
$$= 0.9 \cdot 0.79 \cdot 157 + 1.0 \cdot 1.0 \cdot 240 = 351.63$$

This computation incorporates the level of confidence (*Conf*) of the relevant responding communities, the Matching Level (*ML*) of each internal attribute that is mapped to Att1 (see Table 4), and the support (*Supp* of the inquired subject in the responding communities). The computed certainty is also a part of the CCR score computation, specifically its usage is in normalizing the computation:

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

$$Att1.Score = \frac{Rooms_{(TA)}.Score \cdot Rooms_{(TA)}.ML \cdot TA.Conf \cdot TA.Supp}{Att1.Certainty} + \frac{Comfort_{(Booking)}.Score \cdot Comfort_{(Booking)}.ML \cdot Booking.Conf \cdot Booking.Supp}{Att1.Certainty} = \frac{0.75 \cdot 0.9 \cdot 0.79 \cdot 157 + 0.88 \cdot 1.0 \cdot 1.0 \cdot 240}{351.63} = 0.84$$

The scores used in this computation are already converted to RE (see Formula 5). For simplicity, we disregard the scaling of reputation scores by statistics (we assume a normal distribution in all the communities). We compute the CCR certainty and score for the rest of the relevant generic attributes in a similar manner:

Att2.Certainty = 364.03, Att2.Score = 0.89
Att3.Certainty = 24.81, Att3.Score = 0.75
Att4.Certainty = 351.63, Att4.Score = 0.85
Att5.Certainty = 364.03, Att5.Score = 0.82

Using the calculated certainties and scores for the generic attributes, the corresponding data can be now computed for the internal attributes of the requesting community Expedia. First, the CCR certainty for the inner attribute (for example, *Hotel service*) is calculated:

$$HSer_{(Expedia)}.Certainty = HSer(Att4).ML \cdot Att4.Certainty + HSer(Att5).ML \cdot Att5.Certainty = 0.5 \cdot 351.63 + 0.4 \cdot 364.03 = 321.43$$

The computed certainty is also a part of the CCR score computation, specifically its usage is in normalizing the computation:

$$HSer_{(Expedia)}.Score = \frac{Att4.Score \cdot HSer(Att4).ML \cdot Att4.Certainty + Att5.Score \cdot HSer(Att5).ML \cdot Att5.Certainty}{HSer_{(Expedia)}.Certainty} = \frac{0.85 \cdot 0.5 \cdot 351.63 + 0.82 \cdot 0.4 \cdot 364.03}{321.43} = 0.84$$

We compute the CCR certainty and score for the rest of the internal attributes in a similar manner:

$HCon_{(Expedia)}$.Certainty = 19.85, $HCon_{(Expedia)}$.Score = 0.75
$RCle_{(Expedia)}$.Certainty = 364.03, $RCle_{(Expedia)}$.Score = 0.89
$RCom_{(Expedia)}$.Certainty = 351.63, $RCom_{(Expedia)}$.Score = 0.84

After the CCR computation is complete, a query about the Enterprise Hotel within the Expedia website can produce the output presented in Table 5. Such an output is subject to the display and privacy policies of the website. The right column presents the result of combining the CCR scores with the local reputation of the hotel within the requesting community. This is done by adding Expedia to the list of responding communities. The CCR certainty of most of the attributes is high (over 350), especially when compared to the certainty of the inner score, which is exactly the support of Expedia (19). Consequently, the CCR score in these attributes is the main contributor to the combined score. However, for the Hotel condition attribute the CCR certainty is relatively low (19.85), which makes the local reputation more influential in the computation of the combined *Hotel condition* score.

Table 5: Scores of H1 including CCR in Expedia

Attribute	Local Reputation	CCR	Combined CCR
Hotel Service	4.6	4.2	4.2
Hotel Condition	4.7	3.8	4.2
Room Cleanliness	4.7	4.5	4.5
Room Comfort	4.5	4.2	4.2

Using the additional data enabled by the usage of CCR, the users in Expedia can see that the Enterprise Hotel is in fact not as good as the deficient local reputation in Expedia shows. The hotel's local reputation in Expedia is based on only 19 ratings, while in the other websites it is based on over 100 ratings each. Another important usage of CCR is in bootstrapping the reputation data in new communities or for new hotels in existing communities. As an example, if Booking.com decides to start working with the Ripamontidue Hotel, the hotel can get an initial reputation in Booking.com even without a single rating entered in the website.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

5 TRIC: Architecture Framework for CCR

The TRIC framework defines the basic components and features that are required in order to enable the CCR model functions.

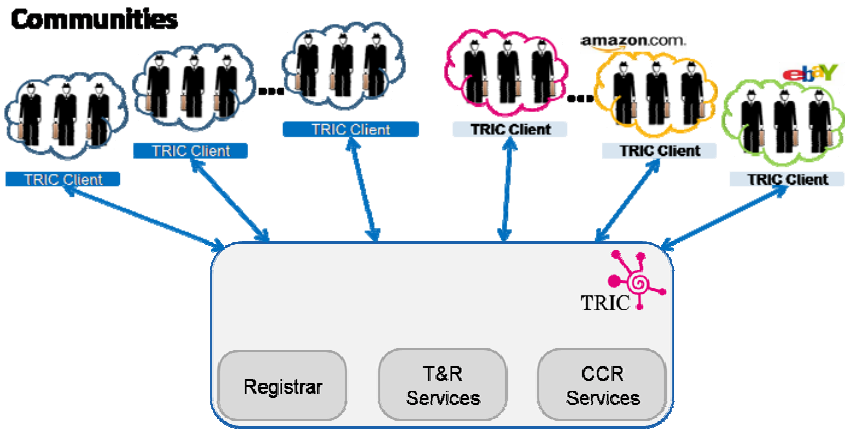

Figure 3: TRIC Services

At the core of the TRIC infrastructure there are three fundamental services: *Registrar*, *T&RP* (Trust and Reputation Provider) and *CCRP* (Cross-Community Reputation Provider). The *Registrar* provides registration services for both users and TRIC clients. The *T&RP* provides reputation computation services for obtaining users' internal (local) reputation within a community. The *CCRP* provides registered communities with cross-community reputation of their users. While the CCR model (section 3) is in the foundations of the CCRP module, the T&RP module implements a trust and reputation model. A trust-based reputation model is essential in the framework of TRIC. A community participating in CCR services should use a trust and reputation mechanism to obtain the reputation of its users. Though we cannot guarantee the robustness of these mechanisms, we require that they address some reliability factors such as the number of experienced events considered (support), the trustworthiness of the input providers (witnesses), the age of the information at hand, etc. Existing communities that already have their own mechanism for computing trust and reputation can continue using it. However, new established communities may choose to use the trust and reputation services provided by TRIC. Currently we use the Knot model as the underlying mechanism of TRIC T&R services. The Knot model [7] is a trust and reputation model for large-scale virtual communities, which was developed as part of the TRIC project.

In the rest of this section we provide a detailed description of the TRIC services (see Figure 3).

5.1 Registrar

The registrar is responsible for all aspects concerning registration and management of users and TRIC clients (communities). It also deals with authentication issues as will be further discussed in section 6.1. The major building blocks of the registrar are depicted at the bottom of Figure 4.

A community that wishes to participate in TRIC, i.e. share reputation knowledge across communities and/or use the T&R services provided by TRIC, should explicitly register to TRIC. The *Client Manager* is in charge of clients' (communities) registration and management. As part of the registration process a community should provide information regarding the attributes it considers as reputation components (usually these are equivalent to the attributes used as a criteria for ranking transactions within the community). Generic attributes are maintained by the *Attribute Manager*. These attributes are derived from the attributes commonly used by the communities. Each community provides a mapping of its attribute in terms of the generic attributes.

As discussed in section 3.1, in order to fully utilize CCR services, a community should provide a set of categories that best characterize it. Generic keywords correspond to categories commonly used by the participating communities to describe themselves (e.g. sports, travel). These categories are either predefined by TRIC administrator or dynamically managed as an ontology within the *Keyword Manager*.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

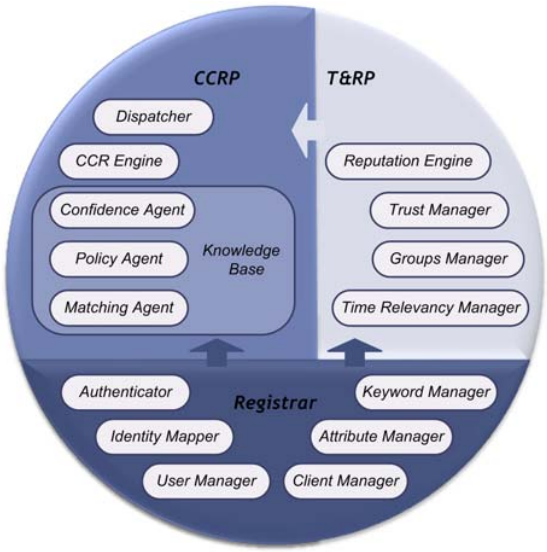

Figure 4: TRIC Infrastructure – Building Blocks

Once a community is registered to CCR services, it can only participate in CCR activities for users that have explicitly expressed their consent by registering to TRIC. A community may redirect a user to register to TRIC but the actual registration is detached from the community. The *User Manager* is responsible for users' registration to TRIC. Registration to TRIC requires the user to provide TRIC with an identity established by an identity provider, e.g. MyOpenID (Site 5). Next, TRIC generates a pseudonym and sends it to the community as the shared pseudonym for further communication. As part of this process the user must give her consent to share her reputation as obtained in one community with other communities she acts in.

When a user is redirected to TRIC from another community, she only has to provide her identification details (the same ones provided when first registered to TRIC). Once again the process is completed by a TRIC-generated pseudonym for further user information exchange with this community. It is important to note that a community has no information regarding the identity that the user provides to TRIC. Correspondingly, TRIC has no information regarding the identity of the user within the community. The *Identity Mapper* module of the registrar manages the user's pseudonyms created for the different clients (communities). The *Authenticator* is in charge of authentication users with their existing identities from supported identity providers, e.g. OpenID (Site 9) providers.

Note that the result of the above design is complete unlinkability between communities [17]. This means that no community will know the identity of a user in another community, nor TRIC will know the nick-name used by a community. Nevertheless, the single identity used within TRIC enables TRIC (or a legal entity using TRIC) to follow activities of abusers across communities.

5.2 CCRP Module

The CCRP module provides cross-community reputation computation services for TRIC clients. The cross-community reputation represented by a CCR object is not unique for every member of TRIC. Two requesting communities may get different CCR objects for the same member even if these CCR objects were compiled based on input from the very same set of responding communities. This is due to potential differences in the confidence that the requesting communities have in each of the responding communities, in the CCR policies of the requesting communities, and in the policy of the member at subject.

The CCR Object is composed of the CCR final score and may additionally include a drill down view of the reputation components. If no restrictions are imposed, the CCR object contains information regarding the responding communities, the reputation objects received by each community including the values of each reputation attribute and possibly textual comments. However, this information is subject to the consent of both the member being the CCR subject and the responding communities.

The CCRP Module collects all information required for the CCR calculation, processes this information, and generates the CCR object. The major building blocks of the CCRP module are shown at the top left side of Figure 4.

The *CCR Engine* compiles the CCR object as described in section 3. It performs the final computation of the CCR attributes, calculates the CCR aggregated score, and produces other information as permitted by the CCR policies. The other components of the CCRP provide the CCR engine with the input required. The *Knowledge Base* in

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Figure 4 is a logical group of components that are responsible for collecting and processing data and provide the engine with the information required to conduct the CCR computation. It consists of the Confidence Agent, Matching Agent and Policy Agent, each responsible for a specific aspect of the CCR computation.

The *Confidence Agent* is responsible for updating explicit information from communities/users regarding their confidence in specific communities. This component is also responsible for computing the confidence level of one community in another community based on information provided by the Matching Agent. The *Matching Agent* is in charge of three major functionalities:

- Compute the extent to which one community matches another community based on information gathered from communities regarding their fields of interest (categories) and the domain of their reputation values (e.g. boolean, real)

- Convert reputation values from one domain to another and scale reputation values based on statistical information

- Map the attributes of one community to the corresponding reputation in another community based on the mapping each community provides to the generic set of attributes

A community may define its terms of playing the CCR game through policies. It can explicitly define its level of confidence in other communities. It can also restrict the transparency of the community from a CCR compiled using the data it provides. A community may prefer to provide information anonymously, i.e. in a way that will not enable a CCR viewer to track back and identify the source of information and even the identity of the user at subject. A user may also define her own terms for sharing her reputation data as obtained in one community with other communities. Like the community a user can restrict the transparency of the community, to ensure that her identity in that community remains obscured.

The *Policy Agent* is responsible for managing the CCR policy provided by the users and by the communities and evaluating the valid privacy rules for a specific CCR request.

Finally, the *Dispatcher* is responsible for composing all requests for reputation information from potential responding communities to a set of requests for CCR. This component performs the required optimization to minimize the number of requests/responses transmitted in this process (see section 6.2).

5.3 T&RP Module

The T&RP module provides trust and reputation computation services for communities that use the TRIC T&R services. The T&RP module is independent of the CCRP, however in order to participate in TRIC CCR service, a community (TRIC client) must have some T&R system that obtains the reputation of community members.

TRIC clients can use this module for computing users' reputation within the community. In the TRIC project we use the Knot model [7] as the reputation computation model. There are two alternatives for a community to use the T&RP module:

- Remote – a client pushes rating information to the T&RP which in turn calculates updated reputation values accordingly and submits the results back to the client. This can be done either on-demand or periodically. This alternative involves transfer of possibly large amounts of data.

- Local – TRIC distributes the T&RP module to the TRIC client. The computation is done by directly accessing the community's database. This alternative involves download of T&RP algorithm upgrades when a new version is available.

In what follows we outline the major building blocks of the T&R module considering the local alternative.

The four components of the T&R correspond to essential aspects of reputation systems. The *Time Relevancy Manager* determines the temporal relevancy of each rating by giving it a weight which is incorporated in reputation calculations that are based on that rating. The weight decays with time so that more recent ratings have higher weights. This reflects the fact that newer ratings are more relevant and thus have more influence on the reputation value. The *Trust Manager* is responsible for calculating trust relations among members. In reputation systems the trust a member has in another as a source of information is used to weigh the input provided by that source. In some systems this value is used to filter out untrusted users' input and to calculate transitive trust relations. In the group-oriented T&R systems and specifically in the Knot model this module also calculates a member's reputation within her group of trust. The *Groups Manager* utilizes the relations obtained by the Trust Manager to generate and maintain groups of trust (knots). The *Reputation Engine* calculates the final reputation score of a member based on the information available within the community. When using the Knot model, it calculates the trust of a member in another based on direct experience or on the experience accumulated in her group of trust (her reputation within a

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

knot). In addition, the Reputation Engine calculates the reputation of a member as perceived by the community as a whole. This reputation is submitted as the community's response to a CCR request.

Finally, let us emphasize again that each community may implement its own T&RP and still participate in the CCR computation by following the required API of the CCRP.

6 Implementation Issues

During our work on the TRIC project we have faced several implementation issues. These include the authorization and authentication of the users using TRIC, as well as various solutions for the exchange of data between the communities and the CCRP.

6.1 Authorization and Authentication

A key objective of the TRIC architecture is to achieve a robust system for reputation sharing while preserving the privacy of the reputation subjects, i.e. the users. A member of a community is encouraged to actively participate while keeping full control over her personal information. The reputation of a member within a community is private data that she may decide to share anonymously with members of other communities. To enable this sharing of information across communities, TRIC should meet two essential requirements. First, a community should get the user's explicit consent to share her reputation with other communities. In terms of service level it is unreasonable to require the user's on-line consent following every request for her CCR. However, we do require the user's offline consent signed in advance. This is actually an authorization the user grants TRIC to provide the community with her CCR in other communities, under the restrictions of her policy. This consent can be limited in time and requires pre-established key sharing between the community and TRIC at community registration time.

Next, we have to make sure that the reputation shared will not cause a linkage between a user's identity in one community and that user's identity in another community. When a user registers to a community she provides a virtual identity that can be authenticated either by the community (e.g., user-name and password) or by an identity provider, e.g. MyOpenId (Site 5). When first registering to TRIC the user should use a virtual identity provided by one of the identity providers supported by TRIC. This identity is used in the next times this user initiates registration to CCR services from a different community. The identity provider supported by TRIC and the identity providers supported by a community may be completely separate entities. This gives the system much flexibility and ability to incorporate existing communities into TRIC.

We require that TRIC will not be aware of the user's identity in the client (community), and vice versa we require that the client will not be aware of the user's identity in TRIC. However, it is mandatory that TRIC and the client interact and refer to the same user. We address these issues at the user registration phase. Registration to TRIC, although redirected from the client, should be initiated by the user. The client submits the registration request on behalf of the user and TRIC generates a pseudonym for the user, and passes it to the client (community). From this point on, TRIC and the client use that pseudonym to identify the user. However, pseudonym generation is done only after the user has approved the community's request to register to TRIC and has authorized the data sharing. The sequence diagram in Figure 5 describes the user-registration process based on the OAuth protocol (Site 10) as implemented in the TRIC prototype.

The OAuth protocol (Site 10) is an example for a standard that meets our requirements. It allows the user to grant access to her private resources on one site (TRIC), to another site (the community). By granting access with OAuth the user does not have to share her identity at all. Apart from OAuth there are other protocols that can be used for exchanging user credentials for an access token (e.g., Google AuthSub, AOL OpenAuth, Yahoo BBAuth, Upcoming API, Flickr API, Amazon Web Services API).

Following the initial user registration to TRIC (redirected from the community), the user may access TRIC directly from the community site in the form of a portlet. The use of such a TRIC portlet allows the user to login directly to TRIC from the community site to edit her policies and view her social credentials. This access does not involve the community at all.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

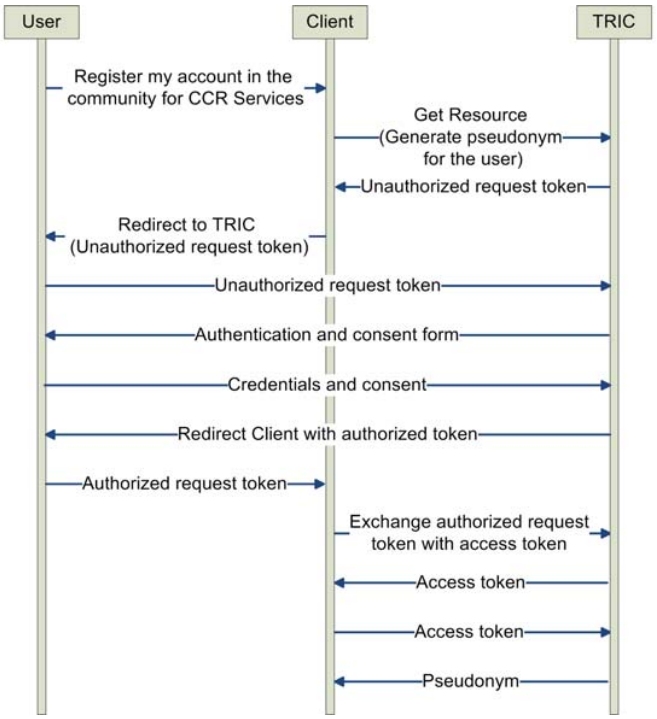

Figure 5: User Registration

6.2 Alternatives for Data Exchange

TRIC resides in a distributed environment, in which the data needed for CCR computation may be spread all over the web. For this reason, data exchange is one of the core issues in the implementation and realization of TRIC. The question of data dissemination has been studied for quite some time (e.g. [2]). Nevertheless, the unique properties of reputation data in the context of CCR call for the design of data exchange alternatives that are designated for this problem. We will classify three aspects regarding the CCR data updates and discuss the alternatives for each aspect. The aspects are – initiator (push/pull), trigger (on-demand/periodic), and sensitivity (any change/threshold).

Initiator – this aspect reflects the identity of the initiator of data updates. The initiator can be either a community or TRIC (specifically the CCRP) depending on the chosen data distribution model:

- *Push* – a community updates the CCRP regarding changes in the local reputations of its users
- *Pull* – the CCRP requests local reputations from a community

The main tradeoff between these two models is in availability *vs.* network load. In the push model the communities are updating the CCRP regarding any changes. Assuming that the CCRP is always available, this means that there is always available data from each of the communities (although the available data may be out-of-date or inaccurate). Contrary to that, in the pull model the CCRP initiates the requests. When it does so the communities might not be available.

With respect to the network load, assuming that changes in local reputation are very frequent, using the push model may overload the network and impair the system's performance. When using the pull model, only required data is exchanged. However, when the changes in reputation data are less frequent than CCR queries the pull model results in a heavier network load than the push model. Thus, in environments that do not have frequent data changes the push model is a better option, especially when considering its better availability.

An additional consideration is the storage of the local reputations. When using the push model, all the local reputation data (of all the users in all the communities) must be stored in a centralized database that the CCRP can always access. However, the pull model does not imply such storage needs. Nevertheless, one may choose to centrally store pulled data for performance increase, e.g. by using cache memory.

The storage issue also effects user privacy and control. Any form of centralized storage (which is a must when using the push model) prevents any possibility for distributed and private CCR computation. Moreover, an untrustworthy CCRP using a centralized storage substantially compromises the privacy of the users. On the other hand, using the

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

pull model with no centralized storage enables the development of distributed methods for private CCR computation. Such methods are out of the scope of this paper (see [9]).

Trigger – this aspect reflects the events that trigger data updates. We consider two strategies:

- *On-demand* – in this reactive strategy the data updates are initiated only when needed
- *Periodic* – in this proactive strategy the data updates are initiated periodically in a predefined frequency

The tradeoffs between these strategies are in data-validity, availability, latency and network load. Choosing to update data on-demand ensures that the data (if available) is valid in the sense that it is up-to-date. When combined with the pull model, the on-demand strategy implies a low network load, but may suffer from availability and latency issues.

When the on-demand strategy is combined with the push model, there are naturally no availability problems. However, the network load is considerably higher, since a community initiates an update following every change (or threshold breach, depending on the sensitivity used) in the local reputation of one of its users.

Another option is to use periodic updates that follow a predefined frequency. The choice of the update frequency leads to a clear tradeoff between data-validity and network load.

Sensitivity – this aspect reflects the amount of change in local reputation that yields a data update:

- *Any change* – any change in the local reputation yields a data update
- *Threshold* – only a change in the local reputation that surpasses a predefined threshold yields a data update

The tradeoff of sensitivity is between accuracy and network load. There are actually not two options here, but rather a continuous function of the threshold value that reflects the sensitivity of update initiations. This is because the any change option can be considered as a threshold of $\varepsilon > 0$. The use of a low threshold value results in accurate data, but with the price of a heavy network load. On the other hand, using increasingly higher threshold values lowers the network load at the price of impaired accuracy.

Note the difference between accuracy and data-validity. Non-valid data means that the reputation value may no longer be relevant due to extreme changes in local reputation data that the CCRP is not yet informed of. On the other hand, the level of accuracy bounds the possible error in the reputation data. Thus, there is more control over inaccurate data than over non-valid data.

Each of the discussed aspects implies a tradeoff between system performance (i.e., network load) and some measure of quality (e.g., data-validity, accuracy). There might also be storage and privacy implications. We next present several combinations of these aspects that suit different needs.

Push-OnDemand-AnyChange – this combination may be used when the main requirement is the quality of the CCR reputation, i.e. availability, data-validity, and accuracy. The cost is clearly in the system performance as well as in reduced user privacy. Any change in local reputation is immediately pushed by the community. Consequently, the CCRP always holds available, up-to-date, and completely accurate reputation information. However, any change in any community is immediately sent to the CCRP causing an extremely heavy network load. Moreover, the CCRP has to store all the data centrally, causing reduced user privacy.

Pull-OnDemand – this combination may be used when the main requirement is maximizing user privacy. Naturally with such a requirement at hand, no caching of reputation data may be performed by the CCRP. This solution may lead to availability problems. If CCR quarries are very frequent this solution also implies a heavy network load.

In the general case there should be some balance between performance, quality, and privacy needs. We therefore propose the following hybrid alternative.

Pull-PushNotify-Cache – each community notifies the CCRP when one of its members' reputation has significantly changed (surpassed a threshold). The CCRP collects these notifications (as well as their respective change ratios). When needed, a community requests CCR data on-demand, and the CCRP pulls only the data it needs based on the notifications it got. The CCRP may also cache recent reputation values, so some redundant pulls may be saved. This alternative does require some storage, but it is much smaller than the storage required for complete storage of reputations.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

7 Experimental Evaluation

The effectiveness of the CCR model was evaluated using two different experiments. In the first experiment we used a sample of real-world user's ratings in the form of hotel recommendations. This experiment focused on the merits of using CCR when having missing or deficient reputation information. Additionally, we conducted a dedicated experiment with real users that involved new expert communities in various academic fields. Through this experiment we are able to examine the effectiveness of the model on different types of attributes.

Before describing the experiments carried out in this work we provide the notations we use throughout this section:
Local Reputation – the reputation of an entity (user, service) within a community computed using all ratings available (complete) or only a subset of these ratings (deficient). The complete local reputation represents the valid reputation of an entity when there is a sufficient amount of ratings within the community to compute it. The deficient local reputation represents the reputation of an entity within a new community or when that entity is relatively new to the community.
CCR – the cross-community reputation of an entity as compiled for a requesting community based on the complete local reputation of that entity in the other (responding) communities.
Combined CCR – the cross-community reputation of an entity as compiled for a requesting community based on the complete local reputation of that entity in the other communities and on the deficient local reputation of that user in the requesting community.

The first experiment was conducted on four well known communities that deal with hotel recommendations – Expedia (Site 7), Hotels.com (Site 11), Booking.com (Site 8), and Venere (Site 12). In order to evaluate the effectiveness of using CCR, we revealed for each examined hotel only the 5 most recent ratings (deficient local reputation). Next, we calculated the CCR values for this hotel based on complete data from the other (responding) communities. Finally, we compared these CCR values to the deficient local reputation with respect to the complete local reputation in the requesting community. This experiment simulates how the deficient information regarding a new hotel entry (5 most recent ratings) can be boosted with the help of CCR.

For this experiment we concentrated on hotels from two major cities – London and New York City. We searched for hotels that had entries with at least 5 ratings in all four communities. Out of approximately 1000 checked hotels in London we were only able to find 19 hotels that met our constraints. This fact alone proves the necessity of CCR, since even in an extensively rated domain such as London hotels, about 98% of the hotels are missing or have deficient information in at least one of these four key communities. In New York, where the hotel supply is lower (around 600 hotels) leading to denser results, we found 27 hotels that met our constraints. Nevertheless, even in New York less than 5% of the hotels have a reasonable amount of ratings in all four communities.

In order to use a community as a responding one in our experiment we need the average rating per attribute for each hotel in this community. Some communities that do not supply such information, e.g. TripAdvisor (Site 6), were excluded from the experiment. Nevertheless, we found four key communities that met our needs – Expedia (Site 7), Hotels.com (Site 11), Booking.com (Site 8), and Venere (Site 12). In order to use a community as a requesting one in our experiment we need the explicit ratings per attribute for each user review. This information is needed in order to extract the 5 most recent ratings for each hotel. Out of the four communities, only Expedia and Hotels.com provide such information, consequently Booking.com and Venere took only the role of responding communities in our experiment.

Both Booking.com and Venere present the reputation scores as real numbers ranging from 0 to 10, so they use a RE value domain. The requesting communities (Expedia and Hotels.com) both use a DS50 value domain. Thus, no conversion uncertainty is implied, and the domain confidence is 1 throughout the experiment. We also assume full category matching and similar reputation mechanisms. These assumptions are reasonable, since all of the discussed communities are websites that deal with hotel recommendations.

The attributes in both requesting communities happen to be the same – *Hotel service*, *Hotel condition*, *Room cleanliness*, and *Room comfort*. For simplicity, we used these four attributes as the generic attributes in the experiment. Table 6 summarizes the mappings of all the attributes relevant in the experiment. Several attributes that do not map to any of the requesting communities' attributes (e.g. *Hotel surroundings* in Venere, *Value for money* in Booking.com) were excluded.

Table 6: Attribute mapping in the hotels experiment

Generic	Expedia / Hotels.com	Booking.com	Venere
Att1 (Service)	Hotel service	Services (0.2) Staff (0.8)	Quality of service
Att2 (Condition)	Hotel condition		
Att3 (Clean)	Room cleanliness	Clean	Cleanliness
Att4 (Comfort)	Room comfort	Comfort	Spaciousness (0.5) Quietness (0.5)

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Figures 6 and 7 present the Mean Absolute Error (MAE) for each of the attributes in Expedia and in Hotels.com, respectively. The scale of the MAE is different in each of the graphs since the ratings in Expedia are ranging from 0 to 5, while in Hotels.com the ratings range from 0 to 10. For each attribute, the left bar represents the MAE of the calculated CCR scores, while the right bar shows the MAE of using deficient local reputation in the form of the 5 most recent ratings. The MAE is calculated with respect to the complete local reputation in the requesting community. The graphs clearly show the effectiveness of using CCR. The middle bar presents the MAE of the combined CCR scores. This is done by adding the requesting community to the list of responding communities. For almost all the attributes the combined CCR results slightly improve the pure CCR results. The improvement is logical since the local reputation (that is part of the combined CCR computation) is the most reliable reputation information. Nevertheless, the improvement is only marginal due to the low support of deficient local reputation in the experiment (only 5 ratings).

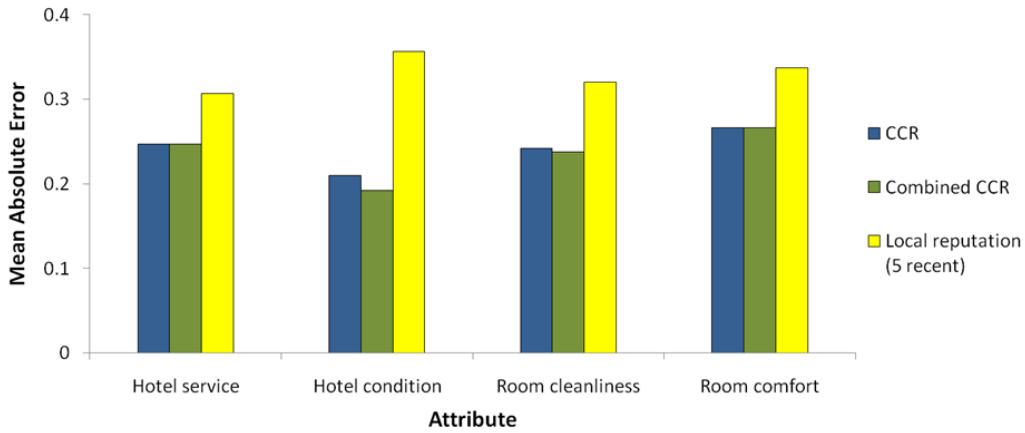

Figure 6: Mean Absolute Error in Expedia

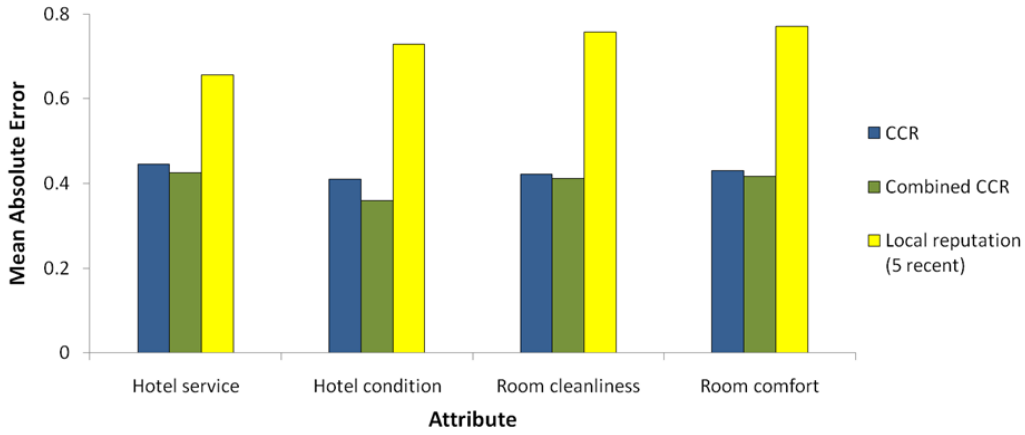

Figure 7: Mean Absolute Error in Hotels.com

An interesting observation is that the 5 most recent ratings in Expedia have slightly less error than those in Hotels.com (disregarding the different scaling). On the other hand, the CCR and combined CCR have lower error in Hotels.com. The reason for this is that the volume of ratings in Expedia is generally lower than in Hotels.com. Consequently, the 5 most recent ratings are much more influential in Expedia, leaving less room for an improvement by using CCR. The good results for the *Hotel condition* attribute suggest that CCR can be effective with even just a single responding community (as Booking.com and Venere have no attribute mapped to the generic attribute Condition).

When deciding on the attribute mapping, one should be careful from similar words that have different meanings. A good example is the *Hotel service* attribute. In Booking.com there is a *Services* attribute that linguistically seems to fit the generic attribute *Service*. However, we know that when a user rates ”the service” she usually means how efficient and pleasant was the staff, rather than which services does the business supply. For this reason we chose the mapping in Table 6 from Booking.com to the generic attribute *Service* to be semantic (*Services*-0.2 and *Staff*-0.8). Figure 8 presents a comparison between the semantic mapping and a linguistic mapping that maps *Services* from

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Booking.com to the generic attribute *Service*. The graph demonstrates the importance of choosing an appropriate mapping. The linguistic mapping even results in a higher error than that of the deficient local reputation in both communities. Nevertheless, due to the non-straightforward mapping of the *Hotel service* attribute, the improvement of CCR is lower than for the other attributes as can be seen in Figures 6 and 7. One may use learning techniques in order to find a more fitting mapping that will reduce the error.

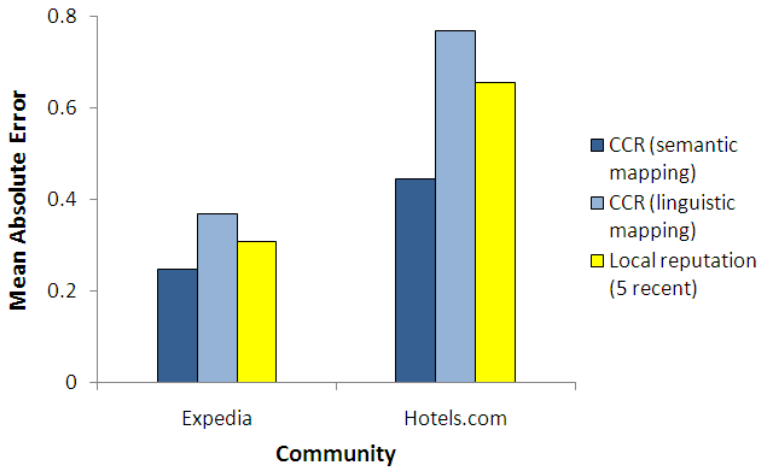

Figure 8: Mean Absolute Error for *Hotel Service* with Different Attribute Mappings

The second experiment was conducted in Ben–Gurion University as part of the TRIC project. The data collected in this experiment was represented as ratings in three virtual communities with experts. Our main purpose in this experiment is to demonstrate the effectiveness of the model on different types of attributes. The three communities are communities of students. The first community involves subjects in Statistics, the second concerns programming in Java, and the third is interested in Information Systems. The same 70 students participate as members of the communities while 10 of them play the role of experts in each of the three communities, and the other 60 interact with them. Each interaction involves a Skype session in which a user asks an expert a question from a pull of questions related to the community at subject. The interaction is anonymous and users rate the experts after each interaction according to four criteria – Clarity, Availability, Politeness, and Proficiency. Rating is a number in the range [1..10] in all three communities. We use the Knot model [7] to evaluate the reputation of each expert in each community. We use 30 randomly selected ratings to compute the expert's complete local reputation in the responding communities and 6 randomly selected ratings to compute her deficient local reputation in the requesting community. We have performed three different sets of experiments. In each set one community plays the role of the requesting community and the other two are the responding communities. We assume maximum confidence between all three communities. This assumption seems reasonable as the subjects of the three communities correspond to courses given by the same department. We distinguished between two types of attributes during the attribute mapping phase – universal and domain-dependent. Universal attributes are features or characteristics independent of the specific context of the community domain such as politeness and availability. Domain–dependent attributes are features or characteristics examined in light of a specific context such as clarity and proficiency. The fact that a member gained high reputation as a professional in one subject can imply to some extent on her proficiency in another subject, only if that other subject is closely related to the former. On the other hand, we expect a person that gained a high reputation as a polite person to be polite in any community she participates in, regardless of the domain at subject. Human behavior is out of the scope of this paper, thus we must state here that the above assumptions should reflect preferences and insights of community managers. A more educated mapping of community attributes may be achieved by equipping community managers with statistical information on past transactions.

We used the following generic attributes for attribute mapping – Clarity, Availability, Politeness, Proficiency-Statistics, Proficiency-Java, and Proficiency-IS. We used attribute matching level of 1 between the generic attributes and the Politeness, Availability and Clarity attributes in each of the communities. Although we consider clarity as a domain-dependent attribute, in this case the experimental results did not show any significant difference between various mapping alternatives. This could be explained by the high category matching level between communities. The proficiency attributes were expressed as three different generic attributes and the selected mapping was based on our understanding that proficiency in Statistics can imply proficiency in IS but not necessarily in Java; proficiency in Java can be related to proficiency in IS but not necessarily to Statistics; proficiency in IS may imply proficiency in both Java and Statistics. The logic behind this selection was the coupling of the subjects as required courses in different departments. The mapping of this experiment is summarized in Table 7.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Table 7: Attribute mapping in the experts experiment

Generic	Community 1(Statistics)	Community 2(Java)	Community 3 (IS)
Att1 (Clarity)	Clarity	Clarity	Clarity
Att2 (Availability)	Availability	Availability	Availability
Att3 (Politeness)	Politeness	Politeness	Politeness
Att4 (Proficiency-S)	Proficiency-S		Proficiency-IS (0.5)
Att5 (Proficiency-J)		Proficiency-J	Proficiency-IS (0.5)
Att6 (Proficiency-IS)	Proficiency-S (0.5)	Proficiency-J (0.5)	Proficiency-IS

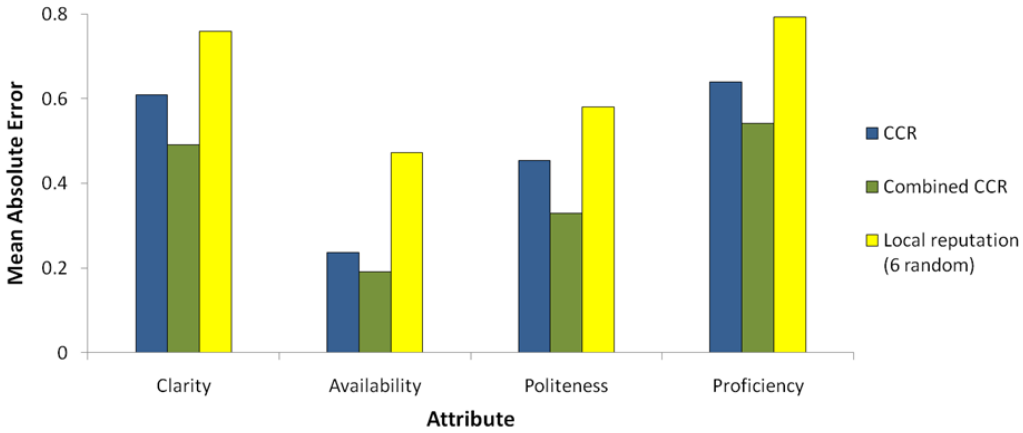

Figure 9: Mean Absolute Error in the Experts Experiment

Figure 9 presents the differences in the mean absolute error of expert reputation prediction with respect to the complete local reputation in the requesting community (based on all 30 ratings). For each attribute, the left bar represents the MAE of the calculated CCR scores, the right bar shows the MAE of deficient local reputation based on 6 ratings, and the middle bar presents the MAE of the combined CCR scores. The results show that universal attributes are clearly easier to predict. Moreover, in all four attributes the CCR has succeeded in reducing the error (by 27% on average), while the combined CCR has even further reduced the error (by more than 40% on average).

Next, we have studied the effectiveness of our model on the domain-dependent attributes, which are harder to predict. In each community, we selected the expert for which the ratings were shown to be most variable. These experts are 6, 9 and 7 for communities 1, 2 and 3 respectively. We examined the complete local reputation of these experts in the domain-dependent attributes and compared them to the CCR compiled using these experts' reputation in the other two communities. The results in Figure 10 show that the prediction error (of the reputation score) is in all cases less than 10% and in 4 out of 6 cases less than 4%. This demonstrates the ability of the CCR model to provide a new community with relevant results when there is nothing else at hand.

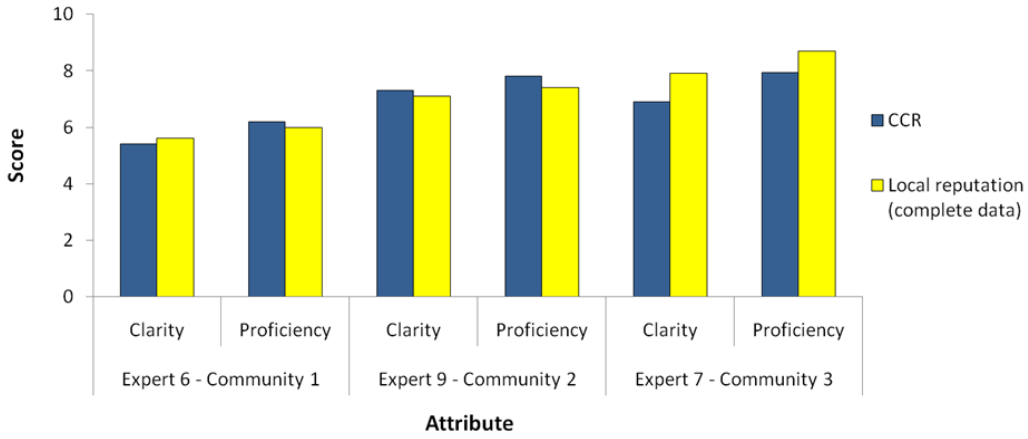

Figure 10: CCR and Complete Local Reputation of Experts with the Highest Rating Variance for Domain-dependent Attributes

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

8 Conclusions

The Cross-Community Reputation model enables sharing of reputation knowledge across virtual communities. The challenge of sharing reputation lies in bridging the differences between the various reputation mechanisms that communities apply. The CCR model facilitates a formal method to convert and assemble reputation data from several communities. The method consists of three main stages – preconditions, conversion of reputation values, and attribute mapping.

The introduced ability of transferring reputation data gives motivation to include reputation as part of a user's identity. With CCR, users can obtain reputation capital [15] in the form of offline reputation certificates that may be viewed as social credentials. The CCR model incorporates privacy policies that users can apply as owners of their reputation data. Using privacy policies users may maintain anonymity or deny undesirable communities from taking part in their CCR computation by *black-listing* them. These features turn the CCR model into a more user-centric scheme. However, they introduce the risk of users manipulating their CCR values. A CCR certificate signed by a trusted TRIC infrastructure can mitigate such a risk.

A precondition for computing CCR is to identify a user that registers to several communities as the same user. This must be done without compromising the user's anonymity in each community and with the requirement of unlinkability between the communities.

The TRIC infrastructure serves as a trusted third party in the current centralized CCR model. We focused on two major design aspects of TRIC – user's control over her data and privacy, and architectural guidelines that can be easily adapted to many environments and standards. We showed how TRIC addresses the first aspect so that a user's reputation in a community is shared and exposed only to the extent permitted by the user itself. The second aspect is brought through the major building blocks that we chose to present as part of the TRIC architecture. These building blocks capture the essence of TRIC on the one hand and demonstrate our open approach for developing such an infrastructure on the other hand.

Two different datasets, both using real collected data, served the experimental evaluation of the CCR model. The first dataset was gathered from hotel recommendations websites. The second dataset was collected by an experiment conducted with students representing virtual communities with experts. The experiments with both datasets demonstrated the power of CCR for communities having missing or deficient reputation information. The importance of good attribute mapping was emphasized in the first experiment while the second experiment showed the effectiveness of our model on different types of attributes. In future use of the CCR model one may apply learning techniques to estimate the attribute matching parameters.

Acknowledgments

This work was partially supported by the Paul Ivanier Center for Robotics Research and Production Management. We thank Dr. Meirav Taieb-Maimon for designing and supervising the experts experiment. We also thank Anat Sirpad and Michal Montal for assembling the hotels data from the different communities and Elena Churkin for implementing and executing the hotels experiment. Our last thank you goes to the TRIC development team lead by Idan Mittelpunkt.

Websites List

Site 1: LinkedIn
http://www.linkedin.com/

Site 2: iKarma
http://www.ikarma.com/

Site 3: TrustPlus
http://www.trustplus.com/

Site 4: eBay
http://www.ebay.com/

Site 5: MyOpenID
http://www.myopenid.com/

Site 6: TripAdvisor
http://www.tripadvisor.com/

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

Site 7: Expedia
http://www.expedia.com/

Site 8: Booking.com
http://www.booking.com/

Site 9: OpenID
http://openid.net/

Site 10: OAuth
http://oauth.net/

Site 11: Hotels.com
http://www.hotels.com/

Site 12: Venere
http://www.venere.com/

References

[1] A. Abdul-Rahman and S. Hailes, Supporting trust in virtual communities, in Proceedings of the 33rd Hawaii International Conference on System Sciences, Maui, Hawaii, USA, 2000, pp. 6007.
[2] M. Bhide, P. Deolasee, A. Katkar, A. Panchbudhe, K. Ramamritham, and P. J. Shenoy, Adaptive push-pull: Disseminating dynamic web data, IEEE Trans. Computers, vol. 51, no. 6, pp. 652-668, 2002.
[3] S. Chakraborty and I. Ray, TrustBAC: Integrating trust relationships into the RBAC model for access control in open systems, in Proceedings of the 11th ACM Symposium on Access Control Models and Technologies, Lake Tahoe, California, USA, 2006, pp. 49-58.
[4] P. Dondio, L. Longo, and S. Barrett, A translation mechanism for recommendations, in Proceedings of the 2nd Joint iTrust and PST Conferences on Privacy, Trust Management and Security, Trondheim, Norway, 2008, pp. 87-102.
[5] N. Gal-Oz, T. Grinshpoun, E. Gudes, and I. Friese, TRIC: An infrastructure for trust and reputation across virtual communities, in Proceedings of the ICIW The Fifth International Conference on Internet and Web Applications and Services, Barcelona, Spain, 2010, pp. 43-50.
[6] N. Gal-Oz, T. Grinshpoun, E. Gudes, and A. Meisels, Cross-community reputation: Policies and alternatives, in Proceedings of the IADIS International Conference on Web Based Communities, Amsterdam, The Netherlands, 2008, pp. 197-201.
[7] N. Gal-Oz, E. Gudes, and D. Hendler, A robust and knot-aware trust-based reputation model, in Proceedings of the 2nd Joint iTrust and PST Conferences on Privacy, Trust Management and Security, Trondheim, Norway, 2008, pp. 167-182.
[8] T. Grinshpoun, N. Gal-Oz, A. Meisels, and E. Gudes, CCR: A model for sharing reputation knowledge across virtual communities, in Proceedings of the 2009 IEEE/WIC/ACM International Conference on Web Intelligence, Milan, Italy, 2009, pp. 34-41.
[9] E. Gudes, N. Gal-Oz, and A. Grubshtein, Methods for computing trust and reputation while preserving privacy, in Proceedings of the 23rd Annual IFIP WG 11.3 Working Conference on Data and Applications Security XXIII, Montreal, Quebec, Canada, 2009, pp. 291-298.
[10] K. Hoffman, D. Zage, and C. Nita-Rotaru, A survey of attack and defense techniques for reputation systems, ACM Computing Surveys, vol. 42, no. 1, pp. 1-31, 2010.
[11] T. Joachims, Text categorization with support vector machines: Learning with many relevant features, in Proceedings of the 10th European Conference on Machine Learning, Chemnitz, Germany, 1998, pp. 137-142.
[12] A. Jøsang and R. Ismail, The beta reputation system, in Proceedings of the 15th Bled Electronic Commerce Conference, Bled, Slovenia, 2002, pp. 324-337.
[13] S. Kamvar, M. Schlosser, and H. Garcia-Molina, The EigenTrust algorithm for reputation management in P2P networks, in Proceedings of WWW2003, Budapest, Hungary, 2003, pp. 640-651.
[14] M. Kinateder, E. Baschny, and K. Rothermel, Towards a generic trust model – comparison of various trust update algorithms, in Proceedings of the 3rd Trust Management Conference, Paris, France, 2005, pp. 177-192.
[15] F. Labalme and K. Burton, Enhancing the internet with reputations, OpenPrivacy.org, Technical Report, 2001. [Online]. Available: http://www.openprivacy.org/papers/200103-white.html.
[16] A. J. Lee and T. Yu, Towards a dynamic and composite model of trust, in Proceedings of the 14th ACM Symposium on Access Control Models and Technologies, Stresa, Italy, 2009, pp. 217-226.
[17] F. Pingel and S. Steinbrecher, Multilateral secure cross-community reputation systems for internet communities, in Proceedings of the 5th International Conference on Trust, Privacy and Security in Digital Business, Turin, Italy, 2008, pp. 69-78.
[18] I. Pinyol, J. Sabater-Mir, and G. Cuní, How to talk about reputation using a common ontology: From definition to implementation, in Proceedings of the 10th Workshop on Trust in Agent Societies, Honolulu, Hawaii, USA, 2007, pp. 90-101.

Journal of Theoretical and Applied Electronic Commerce Research
ISSN 0718–1876 Electronic Version
VOL 5 / ISSUE 2 / AUGUST 2010 / 1-25

This paper is available online at
www.jtaer.com
DOI: 10.4067/S0718-18762010000200002

[19] E. Rahm and P. A. Bernstein, A survey of approaches to automatic schema matching, The VLDB Journal, vol. 10, no. 4, pp. 334-350, 2001.
[20] S. D. Ramchurn, N. R. Jennings, C. Sierra, and L. Godo, Devising a trust model for multi-agent interactions using confidence and reputation, Applied Artificial Intelligence, vol. 18, no. 9-10, pp. 833-852, 2004.
[21] G. Salton, A. Wong, and C. S. Yang, A vector space model for automatic indexing, Communications of the ACM, vol. 18, no. 11, pp. 613-620, 1975.
[22] J.-M. Seigneur and C. D. Jensen, Trading privacy for trust, in Proceedings of the 2nd Trust Management Conference, Oxford, UK, 2004, pp. 93-107.
[23] C. E. Shannon, A mathematical theory of communication, Bell System Technical Journal, vol. 27, no. 3, pp. 379-423, 1948.
[24] D. Strasunskas and S. L. Tomassen, On significance of ontology quality in ontology-driven web search, in Proceedings of the 1st World Summit on Knowledge Society, Athens, Greece, 2008, pp. 469-478.