The VLDB Journal (2009) 18:885–911
DOI 10.1007/s00778-009-0138-1

REGULAR PAPER

RATEWeb: Reputation Assessment for Trust Establishment among Web services

Zaki Malik · Athman Bouguettaya

Received: 25 August 2007 / Revised: 13 December 2008 / Accepted: 20 December 2008 / Published online: 28 February 2009

Abstract We introduce RATEWeb, a framework for establishing trust in service-oriented environments. RATEWeb supports a cooperative model in which Web services share their experiences of the service providers with their peers through feedback ratings. The different ratings are aggregated to derive a service provider's reputation. This in turn is used to evaluate trust. The overall goal of RATEWeb is to facilitate trust-based selection and composition of Web services. We propose a set of decentralized techniques that aim at accurately aggregating the submitted ratings for reputation assessment. We conduct experiments to assess the fairness and accuracy of the proposed techniques.

Keywords Reputation · Trust · Web service

1 Introduction

In recent years, the Web has started a steady evolution to become a "vibrant" environment where applications can be automatically invoked by other Web clients. A key development in this regard has been the introduction of *Web services*. A Web service is a self-describing software application that can be advertised, located, and used across the Web using a set of standards (such as WSDL, UDDI, and SOAP) [49]. Businesses are increasingly using Web services to automate interactions both with their customers (B2C) and amongst each other (B2B). For instance, in B2B interactions Web services are used by businesses to outsource some of the required functionality to other businesses, resulting in the so called "service-oriented enterprise" (SOE) [15,60]. It is expected that enterprises in the new *service Web* would no longer represent monolithic organizations, but rather be a loose coupling of smaller Web-applications offered by autonomous providers [45,49].

The ultimate goal of the Web services technology is enabling the use of Web services as independent components in SOEs that are automatically (i.e., without human intervention) formed as a result of consumer demand and which may dissolve post demand-completion [45]. The service-oriented Web thus represents an attractive paradigm for tomorrow's interactions spanning a wide range of domains from e-economy to e-science and e-government. For example, in e-government, several research prototypes (e.g., WebDG [7,46], WebSenior [44], ARGOS [20]) have shown the viability of the Web service approach in providing e-government services. Similarly, B2B integration through service composition allows services from different providers to be combined into a value-added composite service [49].

On the service Web, Web services will have to automatically determine to which extent they may *trust* other services to provide the required functionality, before they interact with them. By definition, Web services are autonomous (i.e., provided by independent service providers), highly volatile (i.e., low reliability), and a priori unknown (i.e., new or no prior history) [11,45,49]. As a plethora of Web services are expected to compete in offering similar functionalities on the new service Web [70], a key requirement is then to provide mechanisms for the *quality* access and retrieval of services [44,49]. Web services may make promises about the provided service and its associated quality but may fail partially or fully to deliver on these

Z. Malik (✉) · A. Bouguettaya
Department of Computer Science, Virginia Tech,
Blacksburg, VA 24061, USA
e-mail: zaki@vt.edu

A. Bouguettaya
e-mail: athman@vt.edu

A. Bouguettaya
CSIRO-ICT Center, Canberra, Australia

promises bringing down the quality of the whole enterprise. Thus, the challenge lies in providing a framework for enabling the selection and composition of Web services based on trust parameters. The rationale behind the need for trust is the necessity to interact with unknown entities [5,56].

Research results show that reliable *reputation* systems increase users' trust on the Web [14]. For example, several studies attribute eBay's commercial success to its reputation mechanism, known as eBay's Feedback Forum which has been effective in deterring dishonest behavior, and stimulating eBay's growth [21,52]. Similar studies have investigated and generally confirmed that reputation systems benefit both sellers and buyers in e-auctions [26,29]. We anticipate that the deployment of reputation systems on the service Web will have a significant impact on the growth of the different emerging applications such as e-business, e-government, and e-science. Since reputation is regarded as a predictor of future behavior, any Web service with high reputation would be regarded as one that has performed satisfactorily in a consistent manner in the past. This would imply that the service can be trusted to perform as expected in the future as well. In essence, trust is dependent on reputation. Thus, reputation-based trust management will ultimately result in eliminating poor performers and motivating honest behavior among Web services.

In this paper, we introduce RATEWeb: a Reputation Assessment framework for Trust Establishment among Web services. The focus is on providing a comprehensive solution for assessing the reputation of service providers in an accurate, reliable, and decentralized manner. Since reputation forms an integral part of service-oriented environments in relation to the dynamic selection of services, on-the-fly composition of value-added enterprises, and optimization of service tasks, we have chosen Web services as a representative domain. However, RATEWeb can be extended and used in other contexts and domains. The proposed framework takes into account the presence of malicious raters that may exhibit oscillating honest and dishonest behaviors. Previous solutions for reputation assessment make simplifying assumptions that may not apply in a service-oriented environment. For example, Kamvar et al. [25] relies on pre-existing trusted parties, in [6,12] data needs to be distributed according to a certain statistical distribution, a common set of past providers is required in [68] for evaluating rater credibility, and in [42] human intervention is required, meaning the assessment process is not fully automated. Other similar solutions either do not consider all facets of reputation [1,24,54] or are focused primarily on efficiency/performance (rather than functionality) [16]. We develop a simple and holistic solution that provides an automated and adaptive reputation mechanism, whereby reputations are evaluated through a number of heuristics with different perspectives providing a fair and accurate assessment.

The paper is organized as follows. In Sect. 2, we provide a Web services interaction model and highlight the need for a reputation management system with the help of a scenario. Section 3 provides details about our proposed model and the proposed reputation assessment techniques are presented in Sect. 4. Detailed experimental evaluation of our proposed techniques is presented in Sect. 5. This is followed by an overview of the related work in Sect. 6. Section 7 provides some concluding remarks and direction for future work.

2 Web services model

In this section, we present a model of interactions for the service Web. We enumerate the key components of our model and show how these components are related to each other with the help of an example scenario. The scenario also illustrates how reputation is used to establish trust.

2.1 Model entities

Typical interactions on the service Web involve four types of entities: (i) Web services, (ii) service providers, (iii) service registries, and (iv) service consumers.

- *Web services* A Web service is a software application identified by a URI (Uniform Resource Identifier), whose interface and binding are defined, described and discovered by XML artifacts, and supports direct interaction with other software applications using XML messages via Internet-based protocols [45]. Conceptually, a Web service may be viewed as a set of operations, where each operation is a "processing unit" that consumes input values (called its parameters) and generates output values called the result of that operation's invocation. For the sake of focus and clarity, we assume only a single operation per service. The reputation of the operation or the Web service, thus refer to the same thing in our model. However, the RATEWeb approach can be extended to multiple operations per service.
- *Service providers* The service provider is the entity that provides the service, i.e., makes it available to consumers. A service provider may be a business, a government agency, an academic institution, etc. A provider may provide one or more services. A service is provided by a single provider. Providers have publicly known identities. The provider owns the service. It may or may not actually manage the service. For example, the provider of a service may outsource the task of actually operating the service to a third party. Service consumers may or may not be able to discern all the parties involved in delivering a given service. In our model, we do not make a distinction between the service provider and the provided Web service. Thus, when we talk about a service provider, it

is the Web service that is actually provided. The terms service provider, provider Web service and provider are synonymous in our model.

- *Service registries* A service registry is a searchable directory that contains a collection of descriptions of Web services. A service registry has two components: a repository of service descriptions and a registry engine that answers the requests sent to the registry by service providers and service consumers. A service registry may be private or public. Any provider may advertise its capabilities by publishing the Web service in a public registry. A private registry may be used only by a limited, known set of providers to publish services. We focus on the use of public registries in our proposed model. Moreover, we assume no limit on the number of registries. In our model, service registries are only used to locate prospective service providers, and the registries do not store any reputation related information.
- *Service consumers* A service consumer is any entity that invokes a Web service, e.g., an intelligent agent, a Web application, or another Web service. A human user may also invoke a Web service, but we assume that each user is represented by a software component (defined: proxy) in the system. The proxy is thus responsible for all user communication, and managing the functional and non-functional requirements of the user. How this is achieved is not the focus of our work. We assume that the user can *generate* a proxy in one of two ways: (i) implement a custom proxy using a template, or (ii) download a proxy. In privacy-sensitive cases, users may prefer the first technique while the latter provides ease of use. The template or the actual proxy can be obtained through a registry or portal defined by providers that are generally groups of government agencies, non-profit organizations, and businesses that share a common domain of interest. We refer to such providers as "community providers" in our model. Details follow in Sect. 3. We believe the proxy assumption is reasonable as environments that require minimal human intervention (e.g., the Semantic Web [4, 18]) would necessitate the use of such proxies [45]. Without loss of generality, we will assume a symmetric interaction model where typical interactions involve two Web services: one that provides some functionality and another one, the service consumer, that invokes the first one to request that functionality. We also use the terms consumer and client interchangeably to refer to a service consumer.

2.2 Scenario

In this section, we provide a running example to illustrate the need for a reputation management system in a service oriented environment. Consider a car brokerage application (Fig. 1) where a company deploys a *Car Broker* service (CB) that offers a car sale package. "Deployment" of a service means that it is registered with one or more service registries, so that consumers looking to find such a service can obtain invocation details through the registry. For the sake of clarity, we do not show service registries in Fig. 1, and only show the provider–consumer interactions. Note that in our model everything is modeled as a service without much human intervention. A human user that intends to obtain a service, uses a Web service proxy that communicates with the provider service. The human user only communicates his/her needs and preferences to the service proxy, and all decisions

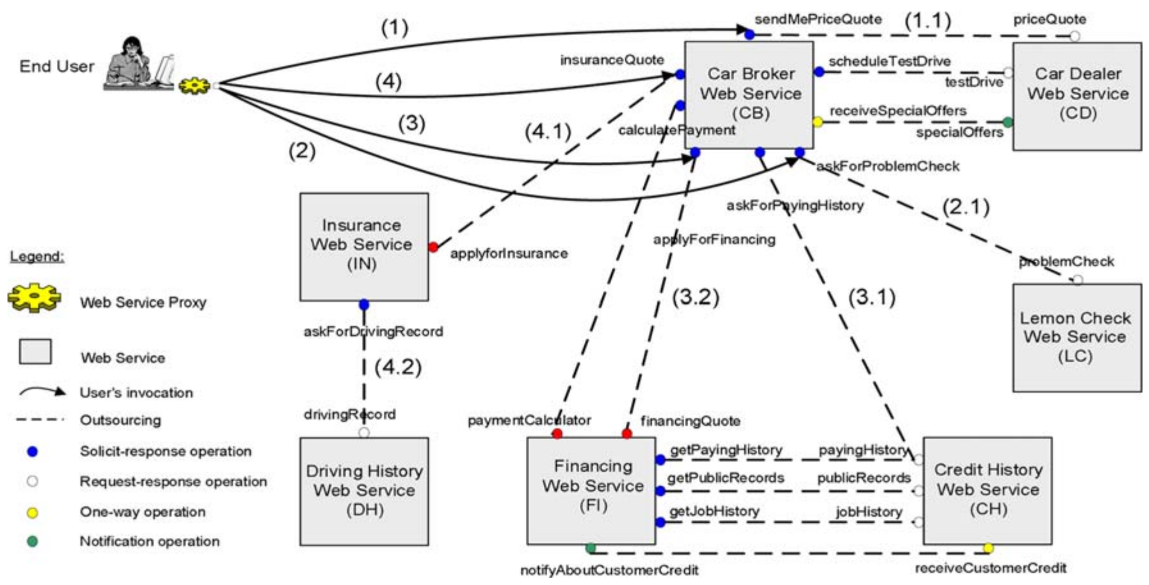

Fig. 1 Car brokerage application

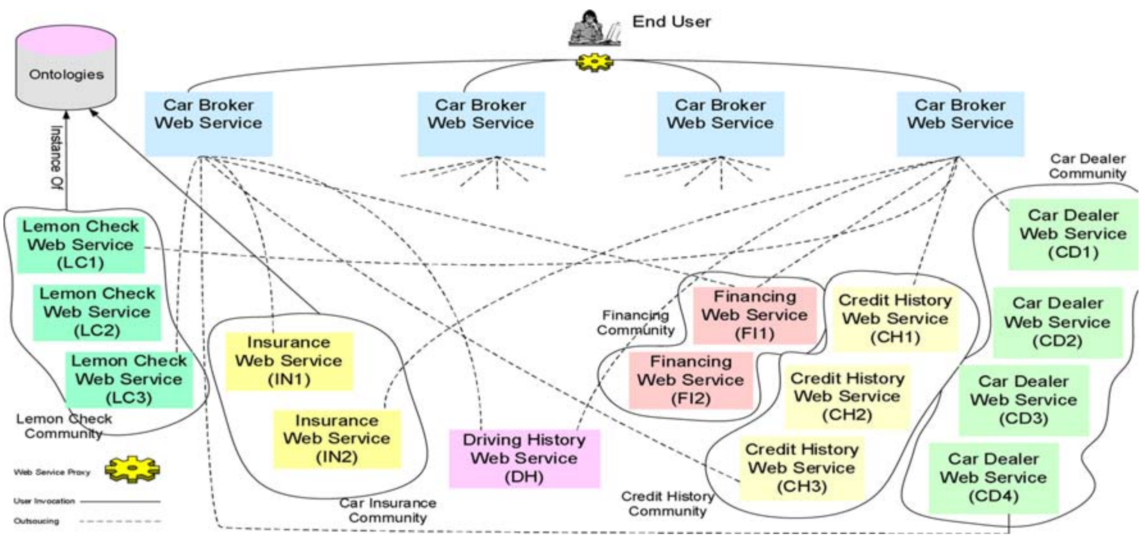
End User
Ontologies
Instance Of
Car Broker Web Service
Car Broker Web Service
Car Broker Web Service
Car Broker Web Service
Car Dealer Community
Car Dealer Web Service (CD1)
Car Dealer Web Service (CD2)
Car Dealer Web Service (CD3)
Car Dealer Web Service (CD4)
Lemon Check Web Service (LC1)
Lemon Check Web Service (LC2)
Lemon Check Web Service (LC3)
Lemon Check Community
Insurance Web Service (IN1)
Insurance Web Service (IN2)
Car Insurance Community
Driving History Web Service (DH)
Financing Community
Financing Web Service (FI1)
Financing Web Service (FI2)
Credit History Web Service (CH1)
Credit History Web Service (CH2)
Credit History Web Service (CH3)
Credit History Community
Web Service Proxy
User Invocation
Outsoucing

Fig. 2 A reputation-driven composition and selection of car brokers

about whom to interact with, how to interact, what steps to take during or after an interaction, etc. are all taken by the service proxy. Thus, in terms of interactions, everything is automated.

To handle the consumer's request, the provider (CB service in our case) may outsource from other Web services, which are also located through service registries. Thus, service providers may also act as consumers. Examples of outsourced services include *Car Dealer* (CD), *Lemon Check* (LC), *FInancing* (FI), *Credit History* (CH), and *INsurance* (IN). These services are provided by different vendors. Moreover, vendors would compete to provide the same (or similar) functionality.

A consumer (user interacting through a service proxy) accesses a CB service to buy a car having a specific make, model, year and mileage. Since a car purchase involves a number of functionalities, a series of invocations of the above mentioned services would need to take place. The selection of a service by CB at each invocation step can be done in two ways. In a use mode where the reputation of the component Web service is *not* considered, the customer would start by invoking CB's `sendMePriceQuote` operation to get a price quote (step (1)). To get a quote, the CB would transparently interact with a *car dealer* via CD's `priceQuote` operation (step (1.1)). If interested in a used car, the consumer would check its history report by invoking CB's `askForProblemCheck` operation (step (2)). This operation is processed by outsourcing from LC's `problemCheck` operation (step (2.1)). The consumer would then apply for financing by invoking the operation `applyForFinancing` provided by CB (step (3)). Before accepting a financing plan, CB would check the consumer's credit by invoking CH's `payingHistory` operation (step (3.1)). If the credit is positive, CB would invoke the `financingQuote` operation offered by the *financing* service (step (3.2)). The consumer would finally request an insurance quote through CB's `insuranceQuote` operation (step (4)). CB would transparently invoke the operation `applyforInsurance` offered by the *insurance* service (step (4.1)). This service would outsource from DH's `drivingRecord` operation before issuing insurance quotes (step (4.2)). Since CB outsources from a number of Web services that are a priori unknown and are located "on-the-fly," no guarantees about the delivery of the required functionality could be made before the actual interaction. This implies that any of the services that CB outsources may exhibit undesirable behavior, effecting the overall consumer experience with CB.

The overall quality of a Web service that the user perceives depends on the behavior of the individual services invoked while answering his request. From the consumers' perspective, the scenario described in Fig. 1 is obviously far from optimal. It does not provide consumers the flexibility to make a quality-based (i.e., using reputation) *selection* of car brokers. Similarly, since the reputation of component services is not considered, any defaulting service may in turn lower the quality of service delivered by CB. In fact, without reputation-based selection, it would be difficult for a service provider to select a *composition* that results in a CB with the "best" possible quality from the perspective of that provider.

Consider now a scenario of a reputation-based selection of car brokers (Fig. 2). In this scenario several companies compete to provide services in the car brokerage business. Each functionality (e.g., checking consumer's credit history) may be satisfied by several Web services. These services would presumably have varying reputations, that may fluctuate over time. In a reputation-aware scenario (Fig. 2), CB providers and consumers are made aware of the reputations of the Web services. The providers may then use services' reputation when *selecting* and/or *composing* their CBs. Similarly, consumers may also select the "best" CB based on the different CBs' individual reputation. Consumers would be able to select only those CBs that have an acceptable reputation, i.e., their past interaction history is satisfactory. Similarly, CB can reduce the risk of its own reputation getting tarnished because of non-reputable outsourced components.

3 Service interactions: extension through ontologies for RATEWeb

In this section we define the RATEWeb framework and show how it extends the existing service interaction model. We present an ontology-based approach for organizing Web services. Ontologies are poised to play a central role to empower Web services with semantics. They are increasingly viewed as key to enabling semantics-driven data access and processing [10]. We introduce the concept of *community* to cater for an ontological organization and description of Web services. A community is a "container" that clumps together Web services related to a specific area of interest (e.g., auto makers, car dealers, etc.). All Web services that belong to a given community share the same area of interest. Communities provide descriptions of desired services (e.g., providing interfaces for *INsurance* services in our running example) without referring to any actual service.

We develop an ontology, called *community ontology*, that serves as a template for describing communities and Web services. A community ontology is a metadata (domain [55,69]) ontology which provides concepts that allow the description of other concepts (communities and Web services in our case). This domain ontology "also describes concept relationships in the application domain, and facilitates the semantic markups on the domain-specific aspects of Web services such as service categories, semantic types of parameters, etc." [67]. Figure 3 outlines the process of creating a community and registering Web services with it. Communities are defined by community providers as instances of the community ontology. Community providers are generally groups of government agencies, non-profit organizations, and businesses that share a common domain of interest. Additional responsibilities of a community provider may include defining a reputation policy that: (i) sets a reputation threshold for members to maintain, (ii) sets rules applicable when a member's reputation goes below the specified threshold, e.g., dissemination within the community of its low reputation, temporary suspension of its membership, and (iii) defining reputation requirements for new members.

A community C_i is formally defined by a tuple (*Identifier*$_i$, *Category*$_i$, *G-operation*$_i$, *Members*$_i$). The *Identifier*$_i$ clause contains a unique *name* and a text *description* that summarizes C_i's features. *Category*$_i$ describes the area of interest of the community. All Web services that belong to C_i have the same category as C_i's. C_i is accessible via a set of operations called *generic operations*. Those are specified in the *G-operation*$_i$ clause. Generic operations are "abstract" operations that summarize the major functions needed by C_i's members. Community providers define generic operations based on their expertise on the corresponding area of interest that is, C_i's category. The term "abstract" means that no implementation is provided for generic operations. Community

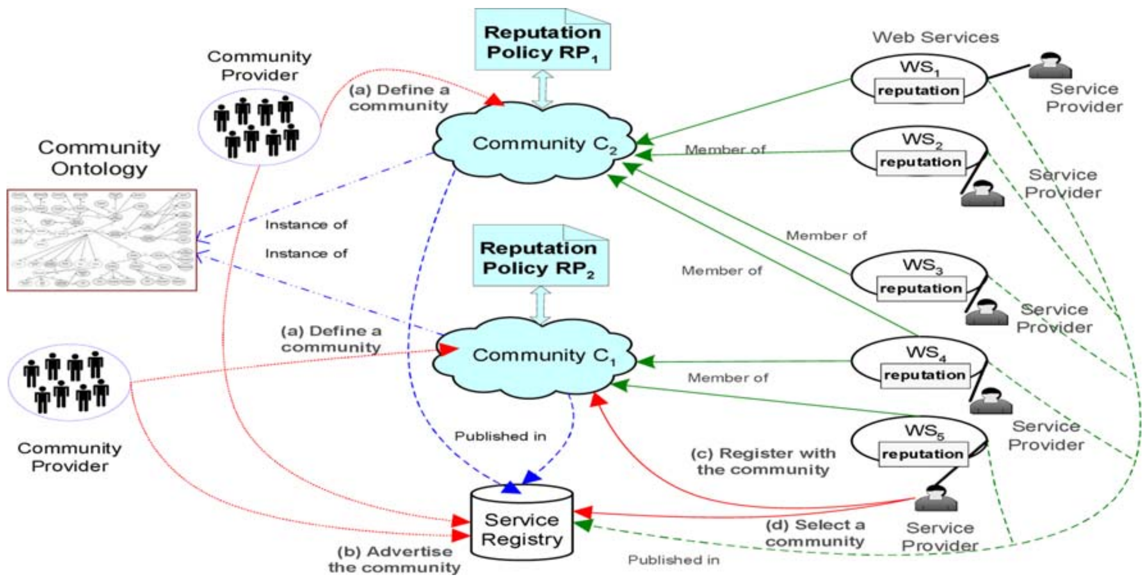

Fig. 3 Community creation and service registration in RATEWeb

providers only define an interface for each generic operation op_{ik}. This interface could subsequently be used and implemented by community members (i.e., actual Web services) interested in offering op_{ik}. We say that those members *support* or *import* op_{ik}. The execution of op_{ik} hence refers to the execution of an actual operation offered by a member that supports op_{ik}. The $Members_i$ clause refers to the list of C_i's members. By being members of C_i, Web service providers promise that they will be *supporting* one or several of C_i's generic operations. Details on communities, the entities and processes involved in the creation, maintenance, etc. thereof, can be found in [44,45,47].

3.1 Model interactions

In RATEWeb, a community is itself a service that is created, advertised, discovered, and invoked as a regular Web service. The providers of a community assign values to the concepts of the community ontology (Fig. 3, step a). Each concept is defined by a set of attributes. Communities are published in a registry (e.g., UDDI) so that they can be discovered by service providers (Fig. 3, step b). Service providers (e.g., `car broker` provider) identify the community of interest (Fig. 3, step c) and register their services with it (Fig. 3, step d). During the registration of a service *WS* with a community C_i, the service provider specifies the concepts of C_i that are inherited by *WS*. For example, *WS* may inherit only some of the operations defined in C_i. Admitting a service to a community is subject to the admission rules specified in the community's reputation policy. Moreover, to make a service available to consumers, a provider publishes the service in a service registry. This description specifies information such as the identity of the provider, the service's address (i.e., URI), its operations, and the number, names, order, and types of each operation's parameters. A service provider may publish the service in one or more registries. For example, a composite service (WS_4 in Fig. 3) may outsource operations that have different domains of interest (e.g., *auto insurance* and *finance* in our scenario). Since these operations belong to two different communities, the composite service is registered with the *auto insurance* and *financing* communities (C_1 and C_2 in Fig. 3). When a service is member of multiple communities, it implies that the service simultaneously fulfills the reputation policy of *all* communities.

Service consumers access service registries to discover the communities and providers of their choice. The consumer's query consists of the operations it wants to invoke. The list of operations is matched with different communities' capabilities. It may be the case that the required operations are matched to several different communities. Each community in turn searches its directory for the list of providers that have registered their operations. It then sends the description of these services to the consumer. The registered services may be invoked to answer the consumer's request. We assume that communities and registries are neutral, i.e., have an impartial policy vis-à-vis the providers of the different services. The service consumer then selects the *best* service from the list provided. In our model, this selection is based on the reputation of each individual service from the list. We assume that when the consumer queries the community for potential providers' list, then apart from the "normal" details, the returned description also contains a list of past service consumers that possess feedbacks for the provider being queried. The community thus only acts as a directory of raters and not as a centralized repository of ratings (ratings are kept local with the raters). The consumer may contact these peer consumers to gather the feedbacks, and in turn assess the providers' reputations. Service consumers then invoke a Web service through one of its listed operations. The consumer provides appropriate values for the operations parameters and the service returns an output value as a result of the invocation. At the end of the interaction, the service consumer *rates* the provider according to some pre-defined quality attributes. The service consumer also informs the community provider that it possesses the feedback ratings for the provider. These service ratings are used to compute provider reputations accordingly. Note that RATEWeb is not dependant on the proposed community-based ratings collection model, and may be replaced with other models, as in [6,44,51,61].

4 Reputation assessment

We view the reputation of a Web service as a reflection of its quality (denoted $QRef$). In this section, we first formally introduce the concept of $QRef$. We then define Web service reputation for non-centralized environments. In the end, we present the reputation metrics used in RATEWeb.

4.1 Parameters reflecting the quality of web services

The "Quality of Service" (*QoS*), is defined as "a set of quantitative and qualitative characteristics of a system necessary to achieve the required functionality of an application" [62]. We adopt this definition of *QoS* and extend its application to the domain of Web services with related constraints, similar to prior works as [19,30,31,38,40]. The quality of Web service ($QRef$) is a mapping between a set of quality parameters and a set of values or ranges of values. Examples include a services' *response time*, *invocation fee*, *availability*, *accessibility*, *reliability*, etc. [30,38]. We assume that mathematical values can be assigned for each exact quality parameter that is included in the model [19,30,31,38,40].

In the context of service-oriented environments, three types of $QRef$ exist: provider-promised $QRef$ ($QRef_p$), consumer-expected $QRef$ ($QRef_r$), and service-delivered $QRef$ ($QRef_d$). The $QRef_p$ values are those that are advertised by the service provider through the service registry. Several models have been proposed for this purpose that range from extending the service registries with $QRef$ information to agent-based frameworks that use ontologies [3, 19,28,30,40,51]. $QRef_r$ represents the preference of the service consumer for each quality parameter. $QRef_d$ represents the actual values that are mapped to the different quality parameters after the consumer interacts with the provider. In other words, $QRef_d$ represents the consumers' perceived or assigned values for each quality parameter. For example, in Fig. 1 the *Credit History* service (CH) may advertise that it is able to provide credit reports of individuals for "last ten years" (i.e., $QRef_p$: credit record of last ten years). A *Car Broker* service (CB) may need to retrieve the credit history of an individual for only the "last seven years" (i.e., $QRef_r$: credit record of last seven years). Since CH's $QRef_p$ offer is available along with the service description, CB can see that CH can fulfill its requirement ($QRef_r$) without actually invoking the service. Assume that when CB does interact with CH, it finds that CH only delivered the "credit record of last three years" (i.e., $QRef_d$: credit record of last three years). Clearly this is unacceptable for CB, and it may not have interacted with CH had it known the true estimate of CH's $QRef_d$. The reputation of CH provides this estimate. Raters can provide their experiences in how much CH's $QRef_p$ and $QRef_d$ differed. If this difference is not large CH is deemed trustworthy, as it delivered what it promised. In contrast, a large difference between $QRef_p$ and $QRef_d$ means CH did not deliver according to its promise, and hence it is untrustworthy.

$QRef_d$ is an approximation of the *actual* quality of the parameters. Many $QRef_d$ parameters will depend on various factors as network traffic, communication infrastructures, etc. Consequently, different consumers may perceive the quality differently even when the provider behaves consistently for all consumers. We assume that consumers agree on the ranges, types, etc. of the values they should assign for each parameter. For instance, the ontologies proposed in [10,40] can be used for this purpose. How different values are assigned to the $QRef_d$ parameters is out of the scope of this paper. Our focus is on using these values in context of reputation. Let S and T be the set of provider Web services and the set of service consumers respectively. Let Φ be the universal set of quality parameters. Φ may be represented as a p-element vector $(\phi_1, \ldots, \phi_p)$ where ϕ_k is the k^{th} quality parameter. When a service requester $t_x \in T$ invokes the service $s_j \in S$, each quality parameter ϕ_k in Φ gets assigned a *delivered quality value* ϕ_k^{xj} (post-transaction completion).

4.2 Web service reputation

RATEWeb's reputation model is distributed in nature. In contrast to third-party-based traditional approaches for reputation management, no single entity is responsible for collecting, updating, and disseminating the reputation of Web services. Each service consumer records its own perceptions of the reputation of only the services it actually invokes. This perception is called *personal evaluation*. For each service s_j that it has invoked, a service consumer t_x maintains a p-element vector $PerEval_j^x$ representing t_x's perception of s_j's behavior. Different strategies may be adopted in updating $PerEval_j^x$. A simple one may be a *per-invocation* update. Upon an invocation of service s_j, the delivered quality $QRef_d$ is compared to service s_j's promised quality $QRef_p$ and, if necessary, a reputation updating algorithm is run to compute the new personal evaluation of service s_j. In essence, personal evaluation reflects the $QRef$ performance of the provider in consumer's views. The personal evaluation $PerEval_j^x$, represents only consumer t_x's perception of the provider s_j's reputation. Other service consumers may differ or concur with t_x's observation of s_j. A service consumer that inquires about the reputation of a given service provider from its peers may get various differing personal evaluation "feedbacks." To get a correct assessment of the service provider's behavior, all the personal evaluations for s_j need to be aggregated.

The aggregation of all personal evaluations to derive a single reputation value is defined as the service provider's *assessed reputation* in that consumer's view. The service consumers may employ different reputation aggregation techniques. Therefore the "assessed reputation" for the provider may be different at each consumer. In light of the feedback-based reputation models, the personal evaluations (as calculated by different service consumers) can be considered as feedbacks, and the assessed reputation as the aggregation of those feedbacks. Note that the notion of assessed reputation as defined in our model differs from the definition of global reputation, in that it is not consistent across all services, i.e., it is an aggregation of all personal evaluations in only consumer t_x's own view.

Definition Let L denote the set of service consumers which have interacted with s_j in the past and are willing to share their personal evaluations of s_j. We assume that L is not empty, i.e., some service willing to share information can be found. Thus, $L \subseteq T$ with $L \neq \emptyset$ and each service x in L has $PerEval_j^x$ values for s_j. Then, reputation of s_j, as viewed by a consumer is defined as:

$$\text{Reputation}(s_j) = \bigwedge_{x \in L} (PerEval_j^x) \tag{1}$$

where $\bigwedge$ represents the aggregation function. Equation 1 provides a first approximation of how the assessed reputation may be calculated. However, the assessed reputation calculation involves various factors that need to be precisely defined and measured.

4.3 Reputation evaluation metrics

RATEWeb is designed in accordance with real world social networks methodologies, which provide better accuracy as they mature, have the ability to evolve, and dynamically evaluate the changing conditions [9]. RATEWeb's metrics are defined to capture most (if not all) aspects of social reputation. The metrics are:

1. *Rater Credibility*: We enable the service consumers to base their decisions according to the credibility of raters. A service consumer's credibility determines how much other service consumers may trust its reported ratings regarding the reputation of the Web services it has invoked. This allows us to differentiate between service trust and feedback trust. For instance, a service that does not have high reputation as a provider (low service trust) may be a credible source (high feedback trust) when it comes to judging the behavior of other service providers, and vice versa. The importance of differentiating between service quality and rating quality has been studied before and it is shown that reputation models that do not differentiate offer little resistance to various reputation attacks [68].
2. *Majority Rating*: We provide a feedback-based reputation system where service consumers can rate the different Web services. The assessed reputation of a service provider is not a mere aggregation but is evaluated on a majority basis.
3. *Past Rating History*: We allow the credibility scores of raters to be updated, based on past ratings history.
4. *Personal Experience for Credibility Evaluation*: We consider the possibility of a rater to default, i.e., provide an incorrect feedback. The consumers can evaluate the honesty of the feedback ratings according to the deviation between their personal experience and the ratings reported by other service consumers (raters).
5. *Personal Preferences*: We provide a personalized reputation evaluation where consumers can weigh the different $QRef$ attributes according to their own preferences.
6. *Personal Experience for Reputation Assessment*: We allow incorporating the "first-hand interaction" data in calculating final reputation scores.
7. *Temporal Sensitivity*: We provide mechanisms to address the temporal sensitivity of ratings, where older ratings are given less weight than present ones.

In the following, we define the above mentioned evaluation metrics in detail. We also show how these metrics help in evaluating an accurate reputation score for Web services. Note that we use *all* the defined metrics in unison to evaluate provider reputations and not in isolation from one another.

4.3.1 Credibility of raters

The foremost drawback of feedback-only based systems is that all ratings are assumed to be honest and unbiased. However, in the real world we clearly distinguish between the testimonies of our sources and weigh the "trusted" ones more than others [59]. A Web service that provides satisfactory service (in accordance with its promised quality ($QRef_p$)), may get incorrect or false ratings from different evaluators due to several malicious motives. In order to cater for such "bad-mouthing" or collusion possibilities, a reputation management system should weigh the ratings of highly credible raters more than consumers with low credibilities [13,22,50, 57,68]. In RATEWeb, the reputation score of the provider is calculated according to the credibility scores of the raters (used as the weight). Thus, Eq. 1 becomes:

$$\text{Reputation}(s_j) = \frac{\sum_{x=1}^{L}(PerEval_j^x * C_r(x))}{\sum_{x=1}^{L} C_r(x)} \tag{2}$$

where Reputation(s_j) is the assessed reputation of s_j as calculated by the service consumer and $C_r(x)$ is the credibility of the service rater x as viewed by the service consumer. The credibility of a service rater lies in the interval [0,1] with 0 identifying a dishonest rater and 1 an honest one. The processes involved in calculating the credibilities of raters are discussed below.

Evaluating Rater Credibility: There are a few existing online systems such as eBay, Amazon, Yahoo! Auctions, etc. that use a centralized reputation system. Most of these systems rely only on the numerical feedbacks received from different users as a reputation measure, or in some cases supplement these with textual feedbacks also left by the consumer. The reputation values are calculated as simple aggregations of the received ratings, which may not accurately predict the trustworthiness of the providers. For example, in eBay (which is one of the most highly used online reputation systems) the buyers and sellers can rate each other on a three point scale, with +1 for a positive rating, 0 for neutral and −1 for a negative rating. The transaction participants are also asked to leave a textual feedback rating. The centralized eBay reputation system then computes the reputation as a summation of all negative and positive ratings received. Since humans are involved directly in processing the provided information (reputation value plus textual feedback), the eBay system has been successful [21,52]. Clearly, such a ratings system is not accurate. A user with 50 positive feedback ratings will have a reputation value equaling one with 300 positive and 250 negative feedback ratings [34]. Moreover, the inability of automated systems to reason in a human-like manner means that the textual feedback will not be of great use. Hence, an eBay-like system may not be practical for the service-oriented environments. Some other online businesses (e.g., Amazon) use an average over all ratings to compute the reputation of a user. Consider a series of ratings: 1, 1, 9, 1, 1, 1, 9, 9, and 1 received for a Web service provider. In an averaging model, the overall reputation score would be 3.7. Clearly, this score is also not in accordance with the ratings received. Thus, designing a ratings system that is robust enough to detect and mitigate the effects of disparate ratings is a fundamental issue [14,66].

To overcome the above mentioned problems, several methods have been proposed in literature that screen the ratings based on their deviations from the majority opinion. Examples include the Beta Deviation Feedback [8], Beta Filtering Feedback [66], Likemindedness [64], and Entropy-Based Screening [65]. We adopt a similar notion to dilute the effects of *unfair* or inconsistent ratings. We use a majority rating scheme, in which the "uniformity of ratings" indicates their accuracy. The basic idea of the proposed method is that: if the reported rating agrees with the majority opinion, the rater's credibility is increased, and decreased otherwise. Unlike previous models, we do not simply disregard/discard the rating if it disagrees with the majority opinion but consider the fact that the rating's inconsistency may be the result of an actual experience. Hence, only the credibility of the rater is changed, but the rating is still considered.

We use a data clustering technique to define the majority opinion by grouping similar feedback ratings together [14, 63]. We use the k-mean clustering algorithm [32] on all current reported ratings to create the clusters. The most densely populated cluster is then labeled as the "majority cluster" and the centroid of the majority cluster is taken as the *majority rating* (denoted M):

$$M = \text{centroid}(\max(\Re_k)) \quad \forall k$$

where k is the total number of clusters, $\max(x)$ gives the cluster $\Re$ with the largest membership and $\text{centroid}(x)$ gives the centroid of the cluster x. The Euclidean distance between the majority rating (M) and the reported rating (V) is computed to adjust the rater credibility. The change in credibility due to majority rating, denoted by Mf is defined as:

$$Mf = \begin{cases} 1 - \frac{\sqrt{\sum_{k=1}^{n}(M-V_k)^2}}{\sigma} & \text{if } \sqrt{\sum_{k=1}^{n}(M-V_k)^2} < \sigma \\ 1 - \frac{\sigma}{\sqrt{\sum_{k=1}^{n}(M-V_k)^2}} & \text{otherwise} \end{cases} \tag{3}$$

where σ is the standard deviation in all the reported ratings. Note that Mf does not denote the rater's credibility (or the weight), but only defines the effect on credibility due to agreement/disagreement with the majority rating. How this effect is applied will be discussed shortly. There may be cases in which the majority of raters collude to provide an incorrect rating for the provider Web service. Moreover, the outlier raters (ones not belonging to the majority cluster) may be the ones who are first to experience the deviant behavior of the providers. Thus, a majority rating scheme "alone" is not sufficient to accurately measure the reputation of a Web service.

We supplement the majority rating scheme by adjusting the credibility of a service rater based on its past behavior as well. The historical information provides an estimate of the *trustworthiness* of the service raters [57,66]. The trustworthiness of the service is computed by looking at the "last assessed reputation value", the present majority rating and that service consumer's provided rating. It is known that precisely defining what constitutes a *credible* rating is an interesting and hard research problem by itself [68]. However, we have attempted to define the credibility of Web services in a practical manner according to the information available to the service consumer. We define a credible rater as one which has performed consistently, accurately, and has proven to be useful (in terms of ratings provided) over a period of time.

Consistency is the defined behavior of a service that exhibits similar results under standard conditions. We believe that under controlled situations (i.e., other variables being the same), a service consumer's perception of a Web service should not deviate much, but stay consistent over time. We assume the interactions take place at time t and the service consumer already has record of the previously assessed reputations (denoted A), which is defined as:

$$A = \bigsqcup_{t-1}^{t-k} \text{Reputation}(s_j)^t \tag{4}$$

where $\text{Reputation}(s_j)$ is as defined in Eq. 1 for each time instance t, $\bigsqcup$ is the aggregation operator and k is the time duration defined by each service consumer. It can vary from one time instance to the complete past reputation record of s_j. Note that A is not the "personal evaluation" of either the service rater or the service consumer but is the "assessed reputation" calculated by the service consumer at the previous time instance(s). If the provider behavior does not change much from the previous time instance, then A and the present rating V should be somewhat similar. Thus, the effect on credibility due to agreement/disagreement with the last assessed reputation value (denoted Af) is defined in a similar manner as Eq. 3:

$$Af = \begin{cases} 1 - \frac{\sqrt{\sum_{k=1}^{n}(A-V_k)^2}}{\sigma} & \text{if } \sqrt{\sum_{k=1}^{n}(A-V_k)^2} < \sigma \\ 1 - \frac{\sigma}{\sqrt{\sum_{k=1}^{n}(A-V_k)^2}} & \text{otherwise} \end{cases} \tag{5}$$

In real-time situations it is difficult to determine the different factors that cause a change in the state of a Web service. A rater may rate the same service differently without any malicious motive, i.e., accurately (but not consistent with the last reporting). Thus, the credibility of a rater may change in a number of ways, depending on the values of V, Mf, and Af. The general formula is:

$$C_r(x) = C_r(x) \pm \aleph * \Upsilon \tag{6}$$

where $\aleph$ is the credibility adjustment normalizing factor, while Υ represents amount of change in credibility due to the equivalence or difference of V with M and A. The signs $\pm$ indicate that either $+$ or $-$ can be used, i.e., the increment or decrement in the credibility depends on the situation. These situations are described in detail in the upcoming discussion.

We place more emphasis on the ratings received in the current time instance than the past ones, similar to previous works as [8,64–66]. Thus, equivalence or difference of V with M takes a precedence over that of V with A. This can be seen from Eq. 6, where the $+$ sign with $\aleph$ indicates $V \simeq M$ while $-$ sign with $\aleph$ means that $V \neq M$. $\aleph$ is defined as:

$$\aleph = C_r(x) \times (1 - \mid V_x - M \mid) \tag{7}$$

Equation 7 states that value of the normalizing factor $\aleph$ depends on the credibility of the rater and the absolute difference between the rater's current feedback and the majority rating calculated. Multiplying by the rater's credibility allows the honest raters to have greater influence over the ratings aggregation process and dishonest raters to lose their credibility quickly in case of a false or malicious rating. The different values of Υ are described next.

Adjusting Rater Credibilities: Υ is made up of Mf and/or Af, and a "pessimism factor" (ρ). The exact value of ρ is left at the discretion of the service consumer, with the exception that its minimum value should be 2. The lower the value of ρ, the more optimistic is the consumer and higher value of ρ are suitable for pessimistic consumers (this value is inverted in Eqs. 10 and 11). We define a pessimistic consumer as one that does not trust the raters easily and reduces their credibility drastically on each false feedback. Moreover, honest raters' reputations are increased at a high rate, meaning that such consumers make friends easily. On the other hand, optimistic consumers tend to "forgive" dishonest feedbacks over short periods (dishonesty over long periods is still punished), and it is difficult to attain high reputation quickly. Only prolonged honesty can guarantee a high credibility in this case. V, M, and A can be related to each other in *one of four* ways, and each condition specifies how Mf and Af are used in the model. In the following, we provide an explanation of each and show how the credibilities are updated in our proposed model using different values for Υ.

1. The local reported reputation value is similar to both the majority rating and the previously assessed reputation, i.e., $(V \simeq M \simeq A)$. The equality $M \simeq A$ suggests that majority of the raters believe the $QoWS$ of s_j has not changed. The service rater's credibility is updated as:

$$C_r(x) = C_r(x) + \aleph * \left(\frac{|Mf + Af|}{\rho}\right) \tag{8}$$

   Equation 8 states that since all factors are equal, the credibility is incremented.
2. The individual reported reputation rating is similar to the majority rating but differs from the previously assessed reputation, i.e. $(V \simeq M)$ and $(V \neq A)$. In this case, the change in the reputation rating could be due to either of the following. First, the rater may be colluding with other service consumers (raters) to increase/decrease the reputation of s_j. Second, the $QoWS$ of s_j may have actually changed since A was last calculated. The service rater's credibility is updated as:

$$C_r(x) = C_r(x) + \aleph * \left(\frac{Mf}{\rho}\right) \tag{9}$$

   Equation 9 states that since $V \simeq M$, the credibility is incremented, but the factor $V \neq A$ limits the incremental value to $(\frac{Mf}{\rho})$ (not as big as the previous case).
3. The individual reported reputation value is similar to the previously assessed reputation but differs from the majority rating, i.e. $(V \neq M)$ and $(V \simeq A)$. The individual reported reputation value may differ due to either of the following. First, V may be providing a rating score that is out-dated. In other words, V may not have the latest score. Second, V may be providing a "false" negative/positive rating for s_j. The third possibility is that V has the correct rating, while other consumers contributing to M may be colluding to increase/decrease s_j's reputation. Neither of these three options should be overlooked. Thus, the service rater's credibility is updated as:

$$C_r(x) = C_r(x) - \aleph * \left(\frac{Af}{\rho}\right) \tag{10}$$

   Equation 10 states that since $V \neq M$, the credibility is decremented. And to cater for the above mentioned possibilities brought in due to the factor $V \simeq A$, the value that is subtracted from the previous credibility is adjusted to $(\frac{Af}{\rho})$.
4. The individual reported reputation value is not similar to either the majority rating or the calculated reputation, i.e. $(V \neq M)$ and $(V \neq A)$. V may differ from the

majority rating and the past calculated reputation due to either of the following. First, V may be the first one to experience s_j's new behavior. Second, V may not know the actual $QRef$ values. Third, V may be lying to increase/decrease s_j's reputation. The service rater's credibility is updated as:

$$C_r(x) = C_r(x) - \aleph * \left(\frac{|Mf + Af|}{\rho}\right) \tag{11}$$

Equation 11 states that the inequality of all factors means that rater's credibility is decremented, where the decremented value is the combination of both the effects Mf and Af.

In RATEWeb, after each interaction, apart from rating the provider s_j, the service consumer also evaluates the usefulness of the raters that provided a rating for s_j. If the Euclidean distance between the consumer's own experience and V_i (both representing s_j's assessed reputation) falls below a predefined threshold, V_i is deemed useful, otherwise it is not. The usefulness of a service is required to calculate a service rater's "propensity to default," i.e., the service rater's tendency to provide false/incorrect ratings. There may also be cases where raters alternate between being useful and not useful, over a period of time. Thus, to get a correct estimate of the rater's propensity to default, we compute the *ratio* of the total number of times the ratings submission was useful (k) over the total number of submissions (n). This is similar to the manner in which peer recommendations are evaluated for usefulness in "recommender systems" [27,58]. The usefulness factor (u_f) is:

$$u_f = \frac{\sum_{i=1}^{k} U_i}{\sum_{x=1}^{n} V_x} \tag{12}$$

where U_i is the submission where the rater was termed "useful" and V_x denotes the total number of ratings submissions by that service. The rater's credibility (calculated using either of Eqs. 8–11) is then adjusted as:

$$C_r(x) = C_r(x) * u_f \tag{13}$$

We need to make a few observations for the above mentioned techniques. First, the consumer can base his decision only on the information he has (from the past), and the information he gathers (in form of feedback ratings). Second, the credibility of the raters is directly influenced by the number of ratings that are similar to each other, and previously assessed provider reputation. The RATEWeb heuristic emphasizes the "majority rating" where the agreement with the majority results in a credibility (and hence weight) increment. We believe this is a valid assumption as malicious raters are likely to be scattered, and an attempt to gain a majority (through collusion) would prove too costly [63]. Third, even if the large majority in one round wrongfully alters a provider's reputation (and rater credibilities), the consequent rounds will detect malicious rating anomalies. If a large number of raters continue to act maliciously for extended periods of time, then such anomalies are hard to detect as a service consumer cannot decide on the actual honesty of the majority of raters. This is also in accordance with the real life social networks phenomenon [9]. However, the consumer's own personal experience can aid in detecting such malicious raters. This will be shown next. We believe that the strength of our proposed technique lies in the ability of a service consumer to identify malicious raters (either in the present round or in consequent ones), and assess the provider reputation accordingly. The proposed techniques for computing rater credibilities are analyzed in [35,36].

4.3.2 Personalized preferences

Service consumers may vary in their reputation evaluations due to their differences in $QRef$ attribute preferences over which a Web service is evaluated. For instance, some consumers may label Web services with high reliability as more reputable while others may consider low-priced services as more reputable. We allow the service consumers to calculate the reputation scores of the Web services according to their own personal preferences. Each service consumer stores its $QRef$ attribute preferences in a *reputation significance vector* (RSV). Since, service consumers can change their preferences from one transaction to the other, the RSV is submitted with each ratings submission. The service consumers can then choose either to accept the reputation evaluation scores of the raters or compute the scores themselves if they have a different RSV. In the latter case, the rater is asked for the individual $QRef$ attribute values instead of the computed personal evaluations. In this manner, the consumers have the ability to weigh the different attributes according to their own preferences.

Let $\phi_h(s_j, u)^x$ denote the rating assigned to attribute h by the service rater x for service provider s_j in transaction u, m denote the total number of attributes and RSV$_h$ denote the preference of the service consumer for attribute h. Then, the local reputation for s_j as reported by service rater x is defined as:

$$PerEval_j^x = \frac{\sum_{h=1}^{m} (\phi_h(s_j, u)^x * \mathrm{RSV}_h)}{\sum_{h=1}^{m} \mathrm{RSV}_h} \tag{14}$$

4.3.3 Temporal sensitivity

Service consumers expect the service providers to behave in a fair and consistent manner. Reputation scores are directly affected by the consistency of Web services. However, there are situations where all the past data is of little or no impor-

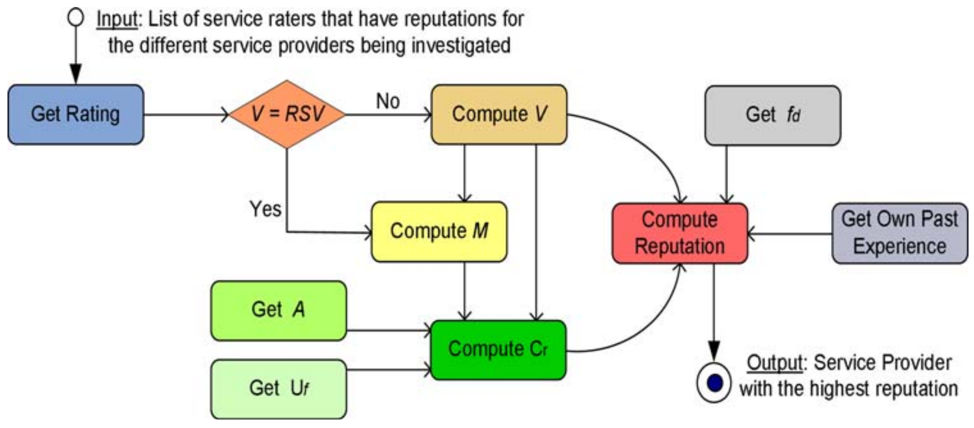
Input: List of service raters that have reputations for the different service providers being investigated
Get Rating
V = RSV
No
Yes
Compute V
Compute M
Get A
Get Uf
Compute Cr
Get fd
Compute Reputation
Get Own Past Experience
Output: Service Provider with the highest reputation

Fig. 4 Reputation evaluation metrics

tance. For instance, a Web service performing inconsistently in the past my ameliorate its behavior. Alternatively, a service's performance may degrade over time. It may be the case that considering all historical data may provide incorrect reputation scores. In order to counter such discrepancies, we incorporate temporal sensitivity in our proposed model. The rating submissions are time-stamped to assign more weight to recent observations and less to older ones for calculating the reputation scores. This is termed as reputation fading where older perceptions gradually fade and fresh ones take their place. We adjust the value of the ratings as:

$$PerEval_j^x = PerEval_j^x * f_d \quad f_d \in [0, 1] \tag{15}$$

where r_i is the rating provided by the service consumer and f_d is the reputation fader. In our model, the recent most rating has the fader value 1 while older observations are decremented at equal intervals for each time instance passed. When $f_d = 0$, the consumer's rating is not considered as it is outdated. The "instance of time" is an assigned factor, which could be anywhere from a single transaction, ten transactions or even more than that. All transactions that are grouped in one instance of time are assigned the same fader value. In this way, the service consumer can define its own "temporal sensitivity degree." For example, one way to calculate the fader is: $f_d = \frac{1}{P_u}$, where P_u is the total number of past transactions over which the reputation is to be evaluated.

4.3.4 First-hand knowledge

Most of the service consumers that have interacted with a Web service provider in the past and were satisfied, continue/prefer to interact with that particular service. Users seldom switch their basic providers online for fear of degrading quality. Web services are inherently dynamic and new services (with better $QRef$) may be introduced in the system any time. Moreover, services with low reputation scores may improve upon their score. However, if service consumers only interact with *trusted* Web services, they may miss better options in terms of $QRef$. We allow the service consumers to incorporate their first-hand interaction knowledge for calculating the final reputation score of the Web services. To the best of our knowledge, present-day reputation systems only allow the users to view/derive a reputation value of the provider based solely on the testimonies of different users. The user's own experience is of a subjective nature which is not factored in the reputation value. Usually, the users do not consider the providers with whom they had a bad experience in the past, even if they receive good reputation scores from other users. In RATEWeb, reported ratings are combined with first-hand knowledge to derive the reputation score. This enables the consumer to consider all Web service possibilities and select the *best* one. Thus, the equation for assessed reputation calculation becomes:

$$\text{Reputation}(s_j) = \frac{\sum_{x=1}^{L}\left[\frac{\sum_{h=1}^{m}(\phi_h(s_j,u)^x * \text{RSV}_h)}{\sum_{h=1}^{m}\text{RSV}_h} * f_d * C_r(x)\right]}{\sum_{x=1}^{L} C_r(x)} \tag{16}$$

Figure 4 shows the pictorial representation of the reputation assessment algorithm that uses the metrics defined above. The input to the algorithm is a list of service raters that have interacted with the service provider(s) in the past and thus have reputation ratings for them. Note that the algorithm iterates over the complete list of potential service providers obtained from a UDDI registry. The output of each algorithm invocation is the service provider with the highest reputation. To simplify the representation, we do not show loops or update processes in Fig. 4.

At the start of the algorithm, a loop is started that iterates over the list of service raters that have a personal evaluation rating (the last "assessed reputation" from their perspective) for the service provider (s_j) in question, and reputation ratings are collected. The service rater returns a vector that comprises of a scalar reputation rating, the RSV of the rater, the ratings for individual attributes and the reputation calculation time-stamp. The RSV of the rater is then compared with the service consumer's own RSV. If the values are similar, the scalar reputation rating is accepted. In case the two RSV's are different, a reputation rating is computed based on the consumer's attribute preferences. This allows the service consumer more flexibility in assimilating reputation ratings from various raters. Thereafter all ratings are used to compute the majority rating. Then, the credibility of the service rater (C_r) is computed by passing the majority rating, the last calculated assessed reputation and the individual rater's rating. Moreover, the credibility is adjusted using the usefulness factor (U_f). The reported reputations (V_i) and each rater's credibility value (C_{r_i}) are then used in computing the "weighted reputation rating" (factoring in user's own past experience, if any). The reputation value is also diluted according to the value of f_d. If this computed reputation value is greater than the reputation value computed for the previous service provider, then the current s_j is labeled as the service with highest reputation value. When all the s_j's reputations have been assessed, the s_j with the highest reputation is identified.

5 Experimental evaluations

We have implemented the above mentioned reputation metrics and associated algorithms to simulate interactions on the service Web. The experiments were conducted in a closed environment where the *actual* behavior of service providers is accurately captured, i.e., we can monitor each service's behavior. The reputations for the service providers are then calculated based on the testimonies of different service consumers. The validity of the proposed metrics is calculated by observing the variance between the actual provider behavior and calculated reputation. In the following, we provide the details of our experiments.

5.1 Setup

We have created a service environment of hundred (100) Web services, with one round of interactions between providers and consumers spanning over 1,000 time-instances. We conduct fifteen rounds of experiments and list the average behavior exhibited. In the current implementation, services are deployed on Windows machines (running XP professional). We developed different *classes* of services (e.g., honest raters, dishonest raters, high performing providers, providers that change quality after a fixed number of transactions, etc.) and then manually created copies of these services to generate the service pool. Different service behaviors in each class are simulated through Java's *randomize* function. For example, to simulate a "high performing" provider we generated its $QRef$ values in the range (0.8–1.0), and a low performing provider in the range (0–0.2). The only data set available (to the best of our knowledge) for measuring QoS of real Web services invocations [2] is used as a guide for generating service provider behaviors.

Table 1 Evaluation parameters

Parameter	Default value	Change value
Number of web services	100	No
Number of transactions in one round	1,000	No
Total experiment rounds	15	No
Quality attributes (denoted $QRef$) measured	5	No
Provider behavior groups	5	No
% of malicious raters (denoted S_m)	50%	Yes (Varies)
% of feedbacks in which raters act maliciously (denoted $rmal$)	None	Yes (Varies)

Using our running example (Fig. 2), we have simulated interactions between *Car Broker* Web services as consumers and *Credit History* Web services as providers. In each time instance, some service consumers and providers interact with each other and at the end of the interaction, service consumers rate the service providers. Also, the two parties record the interaction and digitally sign it. Digitally signing the interaction history helps to avoid the "interaction falsification" attack, where a consumer may provide the rating for a service provider when it has not interacted with that provider. Table 1 shows the list of different parameters used in the experiments. The parameters that are changed during the experiments are mentioned in the column labeled "Change Value," along with the new value. For example, the parameter S_m is changed in some experiments. However, since the new value is not fixed and differs from one experiment to the other, "Varies" is listed as the new value.

5.2 Dynamic provider behavior

Since the presence of malicious providers cannot be discounted, we evaluate RATEWeb against dynamic provider behavior. The hundred service providers are divided into five groups of twenty members each. The groups are created to simulate various malicious behaviors. The first group of providers behave consistently with high $QRef$ values, i.e., these providers behave rationally and do not engage in any malicious activity. The next group performs with consistently low $QRef$ values. These providers are always looking to take advantage of the consumer. The third group performs with high values for the first 500 time instances but then suffer a performance degradation. These are strategic providers that aim to build a reputation by performing honestly initially, and then start "milking" [68] the attained reputation. The fourth group acts in an opposite manner to the

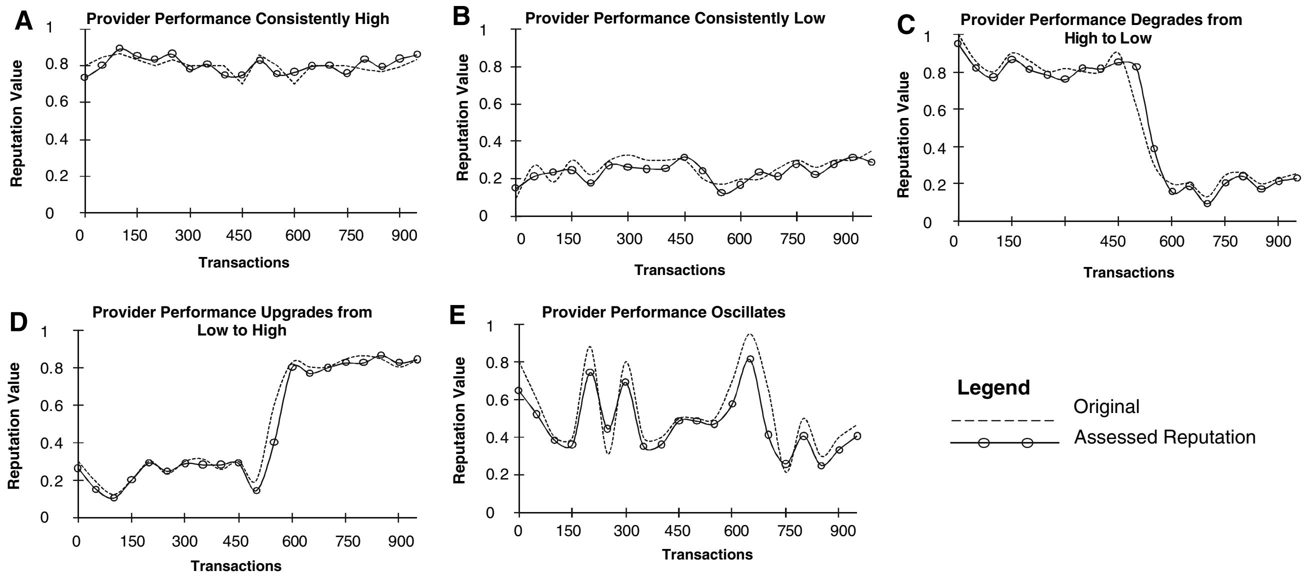

Fig. 5 Reputation assessment when high credibility raters out-number others

third group where providers perform with low values in the beginning. After the 500th time instance, they ameliorate their behavior and start performing with high $QRef$ attribute values. These providers are ones that learn from their mistakes. The final group of providers perform in a random manner, oscillating between high (performing as promised) and low reputation values (acting maliciously). Raters generate their ratings according to the $QRef$ experienced. An honest rater provides the value it experiences, but a dishonest rater generates a rating that differs at least by 0.2 points from the actual rating. Say the provider's $QRef$ value was 0.9, then the dishonest rater would generate a value between (0.1 and 0.69). The five mentioned groups (and any combination thereof) cover any behavior that a service provider may exhibit. This ensures that the experiment samples are representative of the real world environment which contains a variety of provider behaviors.

5.3 Reputation assessment with varying rater credibilities

Since rater credibilities can directly effect the reputation of a service provider, we have also altered rater credibilities to examine the robustness of our proposed methods. The service raters are divided into two groups: honest raters (ones with high credibility), and dishonest raters (ones with low credibility). These groups can be related to each other in one of three ways in the environment: the number of honest raters can exceed those of dishonest raters, honest and dishonest raters can be equal in number, or dishonest raters can out-number honest raters. We set the inequalities in rater behaviors (first and third scenario) to be significant (a 75-25 ratio imbalance is used). In the following, provider reputations are assessed for each scenario and compared with the actual provider reputations.

In the first instance of the experiment, honest raters (ones with high credibility values) out-number dishonest raters. Figure 5 shows the effect of this inequality in calculating the provider's reputation. The plots are labeled A through E to indicate the five provider groups defined above. For instance, Fig. 5a shows the comparison between original provider performance ($QRef_d$) and the assessed reputation for providers that perform consistently with high $QRef$ values. It can be seen that due to the high number of honest ratings, the assessed reputations are almost equal to the original provider performance. The small variation in assessed and original reputations is due to the inconsistency brought in by the (honest) differences in opinions of credible raters and malicious attempts of non-credible raters. This is true for any type of provider behavior (Fig. 5a–e).

The second instance of the experiment where the number of honest and dishonest raters are almost equal in number is shown in Fig. 6. In terms of accuracy, deviation above or below a certain threshold is acceptable. In our experiments, we have set this threshold to be two-points. It can be seen that the assessed reputations are within the ±0.2 range (which is acceptable). Note that selection of the majority rating has direct bearing on the accuracy of RATEWeb. Any rater whose rating is close to the majority rating is deemed honest for that time instance (for which the majority rating is calculated), and dishonest otherwise. When the number of honest and dishonest raters is almost equal, then it may happen that the dishonest raters' ratings form the majority cluster, and hence the majority rating. This causes a degradation in the credibility of honest raters since their opinion now differs from the majority opinion, and an increment in

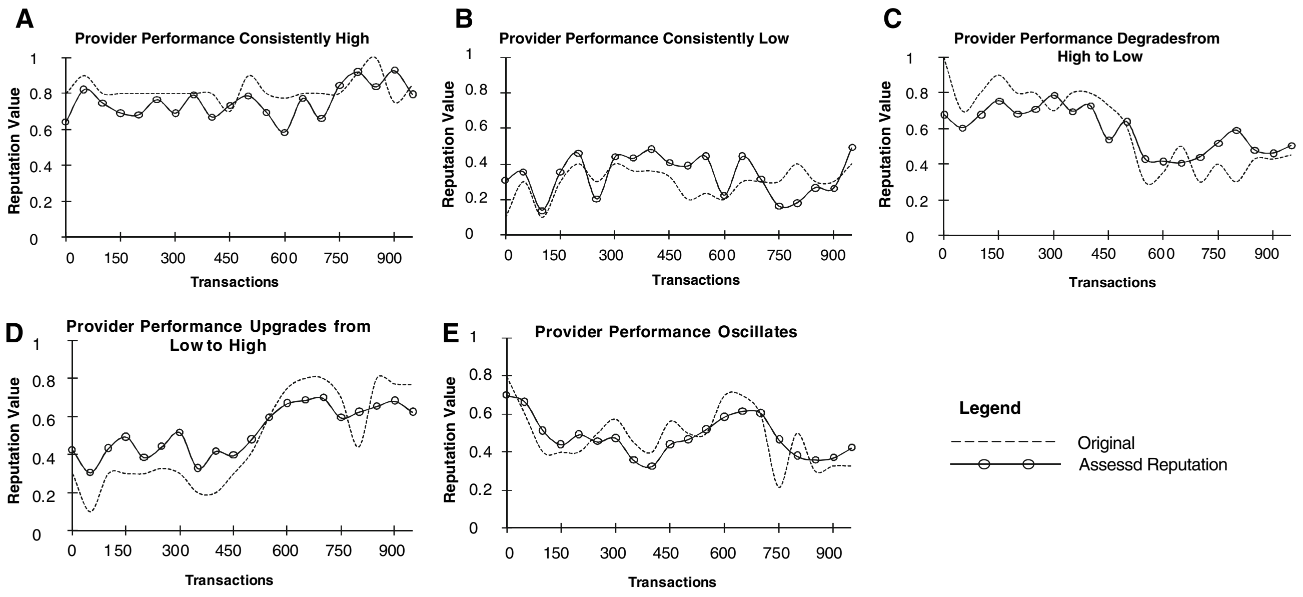

Fig. 6 Reputation assessment when high credibility raters and low credibility raters are equal in number

the dishonest raters' credibilities. Therefore, the majority ratings (and hence reputations) that are calculated in Fig. 6 are sometimes closer to the actual performance (high credibility cluster's centroid chosen as M) and at other times are not so close (low credibility cluster's centroid chosen as M). The rater credibilities *only* get adjusted after the interaction with the provider (through the usefulness factor). This is a known limitation of majority-based systems where if the majority lies then the lie becomes the truth, and honest raters are persecuted.

In the third instance of the experiment, dishonest raters out-number honest raters in the environment. Figure 7 shows the reputations that are calculated in this situation where rater credibilities are mostly low. Comparable to the actual provider behavior, these ratings show some deviation. This is due to the *malicious* reporting of non-credible raters. Note that collusion among raters is not considered for these experiments, and raters agreeing on a (dishonest) reputation value is only incidental. Since consumer reporting is not uniform and is dishonest, assessed reputations show deviation from the actual provider performance values. Still, the assessed reputation are fairly consistent and close to the actual reputations. This is mainly due to the incorporation of first-hand knowledge at the end of each transaction, which dilutes the effects of dishonesty to some extent by lowering (dishonest) rater credibilities. However, the overwhelming majority of malicious raters cause M to change in each time instance, and hence the assessed reputation. This experiment instance shows the "worst-case" scenario of service raters with no collusion. The effects of collusion will be evaluated in the upcoming discussion. Note that in Figs. 6 and 7 at least half of the raters are dishonest. This makes the assessed reputations deviate from the original values. However, in [66], Whitby et al. suggest that such high numbers of malicious raters in real world applications are unrealistic and a much lower rate of dishonesty should be expected. Hence, we may safely conclude that RATEWeb proves to be successful in assessing provider reputations in a fairly accurate manner.

5.4 Adjusting rater credibilities for reputation evaluation

The experiments to this point show the effects of rater credibilities for different provider behaviors. In the following, we list the results of reputation evaluation using different Υ in Eq. 6. Particularly, we use different ρ values to compute rater credibility and consequently provider reputations.

The reputations calculated using low ρ values (optimistic consumer) in Eq. 16 are compared with the original provider performance and the results are shown in Fig. 8. Figure 8a shows the results for a "high performing" provider, i.e., the provider performs consistently with high $QRef$ values. The assessed reputations are shown for two main scenarios. In the first scenario, the majority of raters have high credibility. In the second scenario, malicious raters out-number honest raters. Since low ρ values are chosen, rater credibility suffers low decrement in case of a dishonest rating report. The first scenario results in the calculated reputations being very close to the original provider performance (shown by a dashed line) since dishonesty is minimal. However, in the second scenario, the large number of malicious raters directly affects the majority rating and hence the final assessed reputation. Therefore, the assessed reputation (shown by a dotted line) is not as close to the original performance. Similarly graphs in Fig. 8b–e show the comparison for other service provider

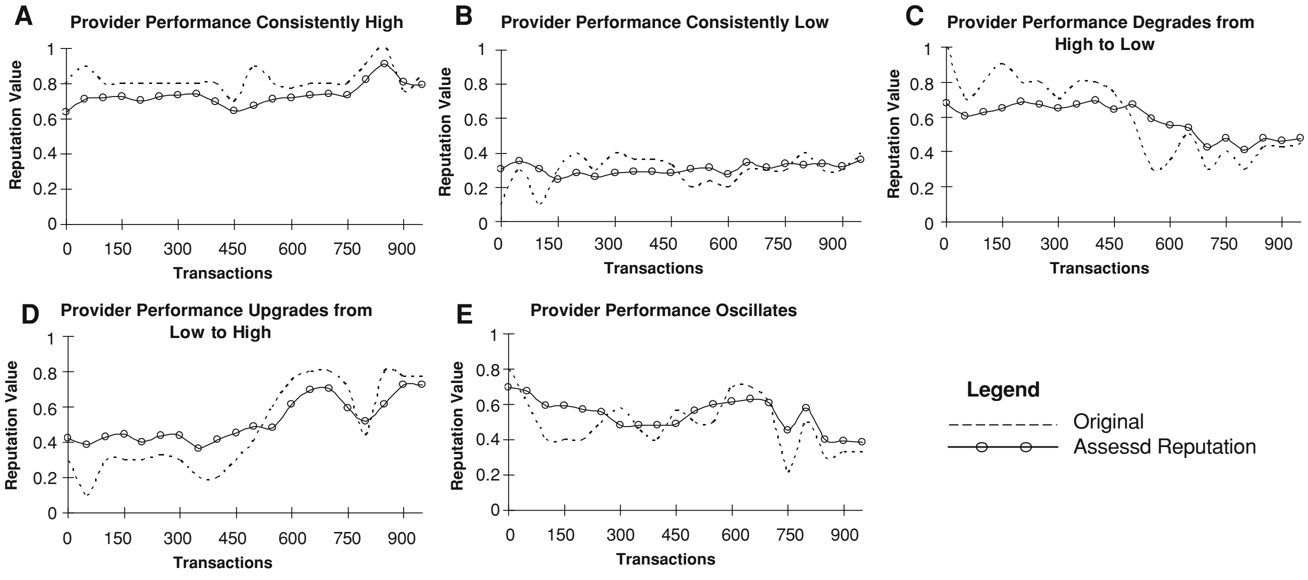

Fig. 7 Reputation assessment when low credibility raters out-number others

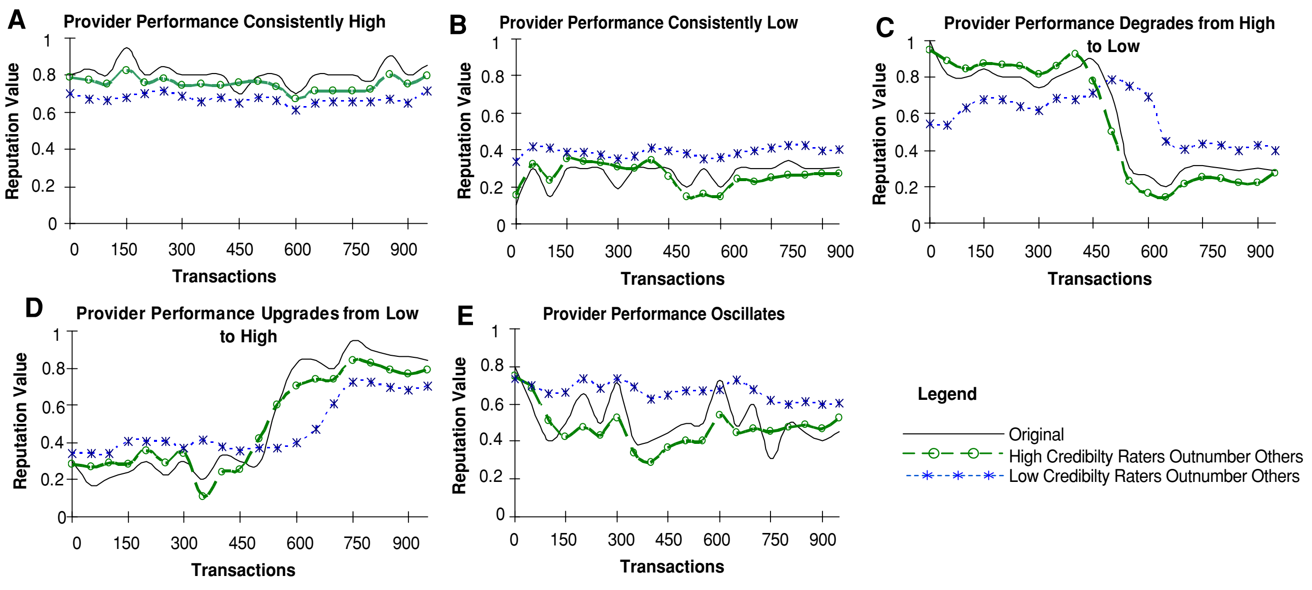

Fig. 8 Original performance and assessed reputation comparison using low ρ value (case of optimistic consumer)

groups. Since dishonest raters are not punished heavily, their credibilities do not fall as much. This causes their dishonest feedbacks in the upcoming transactions to be counted normally, and a little disparity between actual and assessed reputations is observed.

The comparison between original provider performance and assessed reputation using high ρ (pessimistic consumer) values in Eq. 16 is shown in Fig. 9. Figure 9c shows the results for a provider that performs with high $QRef$ values in the beginning and then its performance drops. Similar to the previous case, the assessed reputations are shown for two main scenarios. In the first scenario, the majority of raters have high credibility. In the second scenario, malicious raters out-number honest raters. In our proposed model, any deviation from either M or A negatively effects the rater's credibility. The results in the first scenario are not much different from the previous case and assessed reputations are very close to the original provider performance (shown by a dashed line). This is due to the manner in which credibilities are evaluated. However, the results in the second scenario

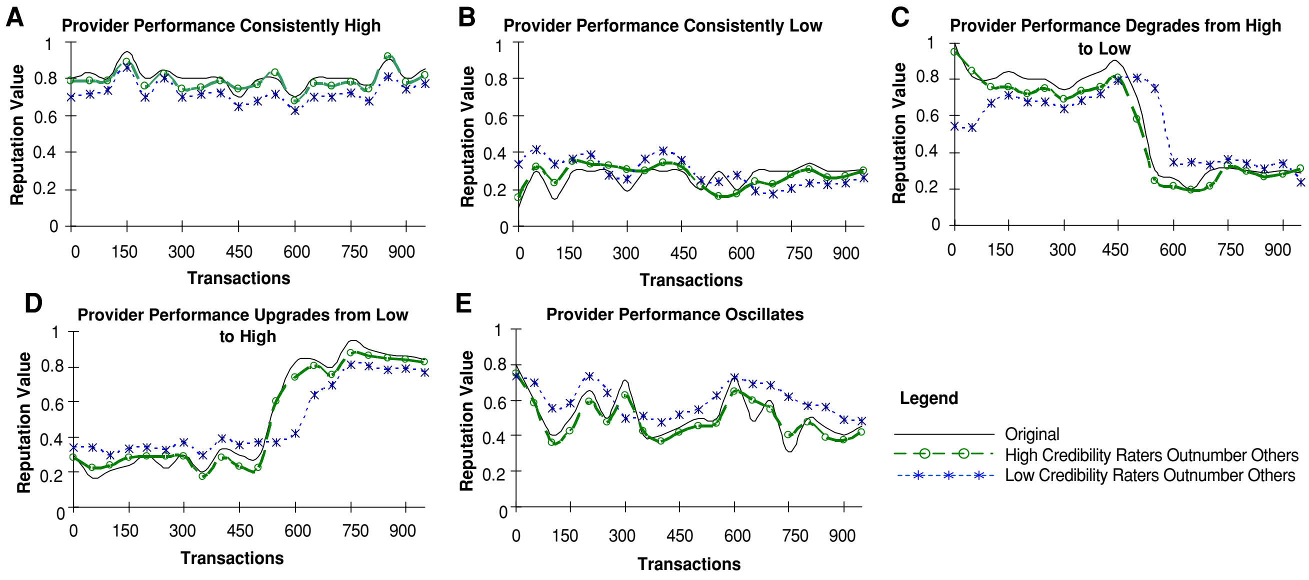

Fig. 9 Original performance and assessed reputation comparison using high ρ value (case of pessimistic consumer)

using a high ρ value differs from the previous case, and the assessed reputations are relatively closer to the original provider performance. This is due to the "punishing" behavior, when rater's evaluation differs from the majority rating and the previous assessed reputation (Eqs. 8–11). The assessed reputations (shown by a dotted line) mostly lie within the two-point threshold, which is an acceptable accuracy. Similarly graphs in Fig. 9a–e show the comparison for other service provider groups.

5.5 RATEWeb comparison

To further evaluate the effectiveness of the RATEWeb metrics, we compare the accuracy of RATEWeb with the conventional approach (in which rater credibilities are ignored and reputations are mere averages of all ratings), and a variant of a popular heuristics-based approach for P2P systems that also considers rater credibilities [68] catered towards service-oriented needs (denoted PeerTrust-V). We model services' malicious behaviors by experimenting under two settings, namely: "with no collusion" and "with collusion". In the setting with no collusion, malicious service providers cheat during transactions and raters provide dishonest ratings. In the collusive setting, malicious services perform similarly to the previous setting, and in addition, collude with other services to increase/decrease some provider's reputation. We change the percentage of malicious raters (denoted S_m) in steps of 10%, and consider a transaction as successful if post-transaction completion, the delivered $QRef$ is close to the computed reputation. Thus, transaction success rate (TR) is defined as the total number of successful transactions over total number of transactions in the community.

Figure 10 shows the effects of changing S_m for the three techniques mentioned above. We can see that since the raters provide dishonest ratings all the time (TR) drops at a consistent rate, and the two settings (with collusion vs. without collusion) exhibit similar results. In the collusive setting, RATEWeb is able to withstand the dishonesty till 50% of the raters are malicious, but the success rate drops thereafter. Since RATEWeb only relies on rater testimonies, when majority of the ratings are dishonest, it becomes difficult for the system to assess the "true" reputation. Incorrect (majority) ratings are considered credible in each time instance and TR drops. PeerTrust-V however is more capable to handle collusion in cases of higher percentages of S_m. In the non-collusive setting, the case is somewhat reversed. With increasing S_m, a large number of ratings may differ from each other which causes raters that deviate from the majority to be labeled as dishonest, diluting the effects of their ratings. Still, with increasing S_m, TR is brought down. Evidence from previous empirical studies of eBay's reputation system (one of the most widely used reputation systems), suggests that more than 90% of the raters stay honest in the system and provide positive ratings to the providers [23,52]. Although such a high percentage may be attributed to the nature of eBay's business model (auction) or its reputation model (both parties rate each other: this was the case at the time of the study. This model has changed recently and only consumers can rate providers now), we believe it provides a rough guideline of the number of credible raters in a ratings-based reputation community. Moreover, as mentioned earlier, it is expected that high numbers of malicious raters in real world applications are unrealistic and a much lower rate of dishonesty should be expected [66]. Thus, we may

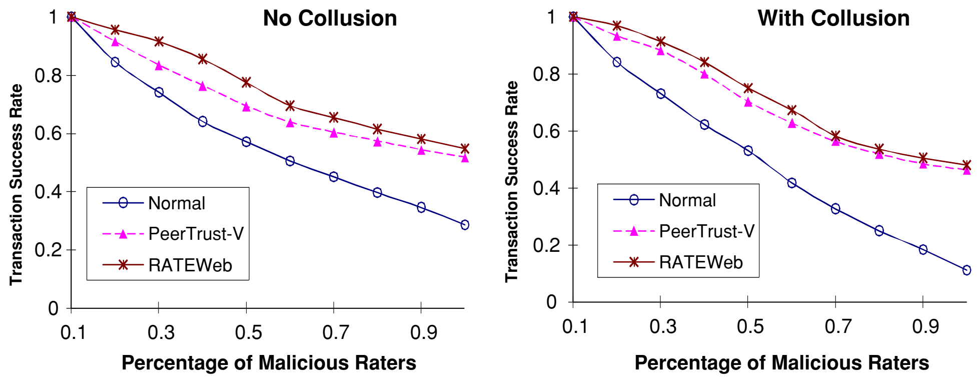

Fig. 10 Transaction success with respect to percentage of malicious raters

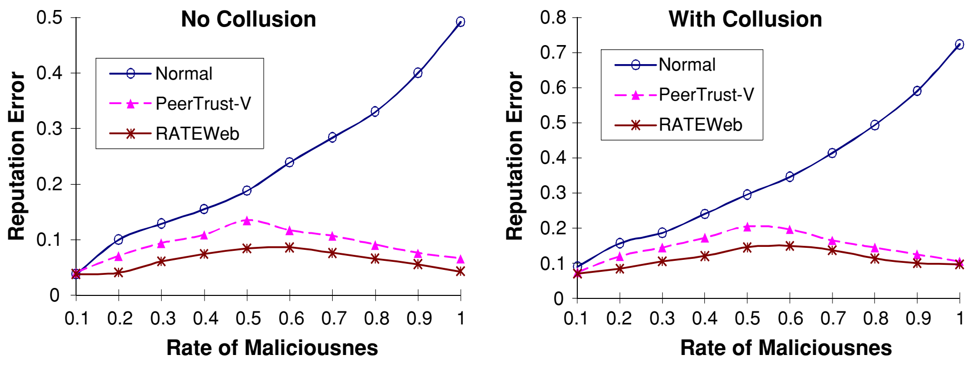

Fig. 11 Reputation accuracy with respect to rate of maliciousness

conclude that in terms of effectiveness for real world applications, RATEWeb provides slightly better results.

We also vary the number of feedbacks in which raters act dishonestly (denoted "Rate of Maliciousness" *rmal*) to elucidate RATEWeb's ability in handling malicious rater behavior. Varying $rmal$ allows us to model the behavior of those raters that attempt to "fool" the system by occasionally providing honest values. Accuracy is measured by estimating the root-mean-square error (RMSE) (denoted reputation error) of the estimated reputations and the actual provider reputations. A low RMSE value indicates better performance. Moreover, we set the percentage of malicious raters in the community (S_m) to 50%.

Figure 11 shows the reputation error comparisons for the different approaches, with variable *rmal*. Both PeerTrust-V and RATEWeb outperform the normal approach, with RATEWeb providing slightly better results than PeerTrust-V. In both settings, the reputation error is around its maximum when *rmal* constitute half of the ratings. This indicates that when less than half of the testimonies are false, the effects of these transactions are overlooked due to the honest majority of ratings. Similarly, when *rmal* exceeds 50%, rater credibility is negatively affected, and false testimonies are filtered out (since half of the raters are honest). Alternatively, when half of the rating testimonies are malicious, raters may be able to confuse the system a little bit more. However, note that in the worst case the reputation error is only 0.17. With S_m expected to be less than 50% [52,23,66], RATEWeb's accuracy is deemed acceptable. RATEWeb's accuracy may be attributed to the way rater credibilities are evaluated in our model [35], and the use of "personalized preferences".

The experiments described in this section show that RATEWeb generates service reputations in a fairly consistent, accurate and effective manner. The process of reputation generation does not create any "spikes" due to the actual behavior inconsistencies and is gradual. The gain and degradation of reputation, are both gradual processes. The proposed techniques inhibit a variety of attacks that include interaction falsification, unfair ratings for both bad mouthing and complimentary purposes, strategic rating falsification, provider behavior deterioration, incomprehensive provider evaluation, and the staleness of ratings. Since the proposed heuristics work *only* with the ratings submitted by different consumers and no trusted third party agents or services are employed in the reputation assessment procedures, the honesty or dishonesty of majority of the raters directly affects a provider's reputation [9]. In the experiments, the consumer ratings were based on the actual provider performance to simulate real-world scenarios. Thus, based on the experimental results we conclude that the proposed RATEWeb metrics can be used for real world applications.

5.6 Cost analysis

The objective of the experiments described in this section is to understand the runtime overhead of deploying RATE-

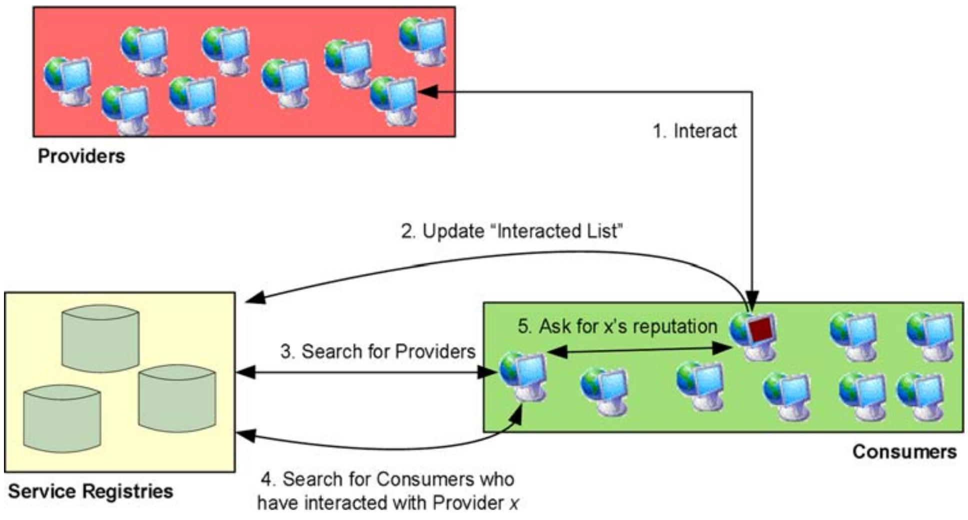
Providers
1. Interact
2. Update "Interacted List"
5. Ask for x's reputation
3. Search for Providers
Service Registries
Consumers
4. Search for Consumers who have interacted with Provider *x*

Fig. 12 Publish-subscribe collection model

Web, to see how well it scales. Runtime overhead mainly involves the cost of retrieving required information (in form of feedbacks) and the time it takes to assimilate all the gathered information. Thus, the total cost is directly influenced by the reputation collection model used. For experimental purposes we define three such models for service-oriented environments and compare their costs. Note that each collection model merits its own extensive discussion, which is not the objective of this paper. In this paper, we are merely using these models to analyze the reputation computation costs. In the following, first a brief overview of each collection model is presented. This is followed by a series of cost analysis experiments.

5.6.1 Publish-subscribe model

In this model, consumers have the ability to publish the list of providers they have interacted with (and are willing to share their interaction experience) in a repository/registry. This allows other consumers to look for raters in regard to a specific service provider. It is assumed that service registries and consumers will have operations defined that facilitate the processes of updating interaction lists and ratings discovery respectively. For instance, similar to previous works that add QoS information in the UDDI along with service descriptions [3,19,28,30,40,51], we could also add the IDs of providers, the consumer is willing to share information (i.e., provide feedback) about. Figure 12 shows the step-wise details of the process. In the first step a consumer interacts with a provider. It then updates its interaction list to include the provider it just interacted with (and holds interaction $QRefs$). When another consumer looks for prospective providers (step 3), it can look for other consumers (raters) that have interacted with those providers (step 4). Since the actual $QRef$ values reside with the rater, the consumer interacts with the rater (step 5) to retrieve the desired rating.

5.6.2 Community broadcast model

In the community broadcast model, we use ontologies to define ratings-communities. Any service consumer that intends to publish or obtain reputation ratings is required to register with a community. This is done on voluntary basis and only registered consumers can obtain and share ratings. At the completion of a service request (consumer–provider interaction), the consumer disseminates interaction $QRef$ values (i.e., rating) in the community. We use a broadcast-based approach in which each registered consumer receives the ratings. Figure 13 shows the step-wise details of the process. In the first step consumers register with the community. After a consumer interacts with the provider (step 2), it broadcasts the interaction ratings in the whole community (step 3). Other service consumers that discover a list of prospective providers through the service registry (step 4), can use the ratings they received in step 3 to assess the provider reputations accordingly (step 5). Note that a major drawback of this model is possibility of "useless" traffic in the network (as services that do not need a provider's reputation get it anyhow).

5.6.3 Credibility-based model

In the credibility-based model, service consumers form rating "cliques" or groups. A service consumer maintains a set of credible raters and requests ratings for a given provider only from the set of credible raters. It may happen that the credible raters do not have the required ratings. In this case, the ratings request is forwarded to the set of credibles for each credible service that does not store the required ratings. The

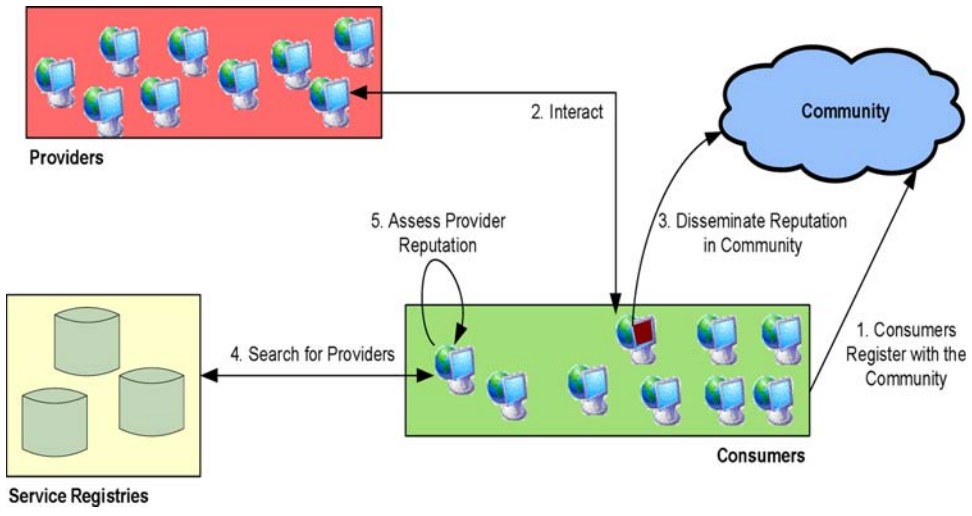
Providers
2. Interact
Community
3. Disseminate Reputation in Community
5. Assess Provider Reputation
1. Consumers Register with the Community
4. Search for Providers
Consumers
Service Registries

Fig. 13 Community broadcast collection model

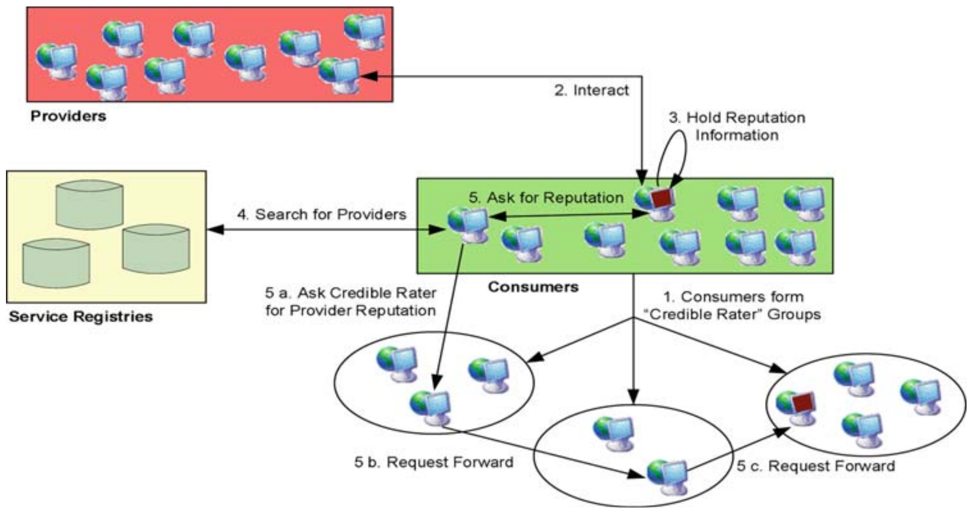

Fig. 14 Credibility-based collection model

requesting consumer can place a limit on the number of hops before a ratings query is exhausted. Figure 14 shows the stepwise details of the process. In the first step, consumers form "credible rater" groups (bootstrapping may be experience-based or through random assignment). After a consumer interacts with a provider (step 2), it does not disseminate the $QRef$ information, but hold it for others to query it (step 3). Other service consumers that discover a list of prospective providers through the service registry (step 4), ask their "credible raters" if they have ratings for the providers (step 5a). If a rater in the group has the rating, it is communicated (step 5), otherwise the ratings query is forwarded by the credible raters to their list of credible groups (steps 5b and c). This is similar to existing P2P architectures with the added constraint that only trustworthy raters are consulted. The major benefit of this approach is that the probability of only getting trustworthy ratings is high. However, a drawback is that ratings may become scarce as it is not highly likely that services in the credible set would have interacted with the provider in question.

We have created a service environment in which the reputation of a single service provider is to be assessed using RATEWeb, from a single consumer's point of view. In the first experiment, we vary the number of service raters from 50 to 1500 and observe the performance of each collection model described above. The service consumer gathers all the ratings and assesses the provider's reputation. We measure the total time (in seconds) RATEWeb takes to assess the reputation of the provider. This includes (i) the rating lookup cost, and (ii) reputation assessment cost. For the former, we generate simple Web services that provide only the rating for the provider once invoked. The ratings are generated in a random manner (since RATEWeb accuracy is not the objective here). For generating the large number of Web services, we use the in-house developed Web Service Benchmark Framework (WSBF) [47].

We have used the default parameter values as listed in Table 2. For the Publish-Subscribe model, we assume that once a consumer gets the list of raters from the registry, it can retrieve the ratings from 20 raters in 1 s. The rater response

Table 2 Performance evaluation parameters

Parameter	Default value
Number of raters	50
Rater response time	75 ms
Registry response time	50 ms
Reputation assessment time for 50 ratings	86 ms
Broadcast time for 50 raters	0.5 s
Publish-subscribe requests/s	20
Depth of credibility groups	4
Number of communities	1

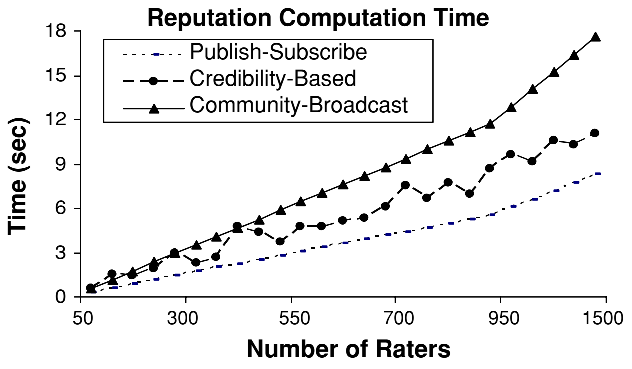

Fig. 15 RATEWeb performance analysis

time to send the ratings is set to 75 ms (when no congestions are present). It is therefore not important in which order the ratings arrive (since all have same time). The rest of the time is spent in processing the ratings (i.e., majority rating calculation, credibility updates, reputation assessment). For instance, in Table 2 the assessment time for 50 ratings is 86ms. In the Credibility-Based model we assume that only four "hops" are required among the credibility groups and that all raters are part of one of these groups. In Table 2, we define this maximum number of credibility groups that can be queried (e.g., if the consumer's credible raters do not possess the rating, it is forwarded to the second group) as the depth of the group. For the Community-Broadcast model we assume that all services belong to a single community (upcoming experiments change this simplifying assumption), and that it takes 0.5 s for the ratings of 50 raters to be disseminated in the community (i.e., reaching all 50 participants).

Figure 15 shows the comparison between the three models. We can see that the Community-Broadcast model exhibits the worst performance. This is due to the fact that as the number of raters increases in the community, the overhead of transmitting the ratings to all participants also increases. The graph for the Credibility-Based model shows variations along the upward trend. We note that in this model, since all raters are not honest, all ratings are not required by the consumer. Therefore, we generate credibility groups for the raters which have the maximum size of one-third of the total number of raters. Once the consumer receives the required number of ratings, it can start the process of aggregation. The variations in the Credibility-Based model are due to the random generation of credible groups which cause the required ratings to be found sometimes at depth 1 or increase to 2, 3 and 4 respectively. The Publish-Subscribe model shows the best performance. Since we do not consider registry or rater bottle-neck effects (retrieving rater identities and ratings respectively), only the time required to access the ratings and aggregating them is taken. Note that the results are for a consumer's ability to retrieve 20 ratings simultaneously, and the performance can be directly effected if this number is changed.

Figure 16 shows three experiments in which we vary different parameters to see how the reputation assessment time is effected. In the first experiment, we vary the number of consumers (in the Publish-Subscribe Model) that are trying to retrieve the ratings for the provider at the same time (Fig. 16a). We simulate simultaneous access by multiplying the number of consumers with the average time for one consumer and adding it to the total time. In real situations this number may be a little low, depending on the multiprocessing abilities of the rater. However, for experimental purposes, we believe that a simple summation of all retrieval times should be acceptable. We can see that the "bottle-neck" effects experienced by the consumer, both at the registry (to obtain rater lists) and at the rater (ratings retrieval) increase the total time experienced by the consumer. In the second experiment, we vary the number of communities, and number of members per community for the Community Broadcast Model (Fig. 16b). Here we assume that the consumer is registered with all available communities (2, 5, 30, etc.), and it starts reputation aggregation only after the ratings are disseminated among all community participants. We can see that it takes about 19 s for ratings to be disseminated among 1,500 participants for a rater that is registered with 30 communities. Although this is a large number, we believe that in real-life situations such a scenario may not exist, and services may register with less than 10 communities at a time (and total computation time for this case is around 6 s). In the third experiment, we vary the depth of credible rater groups, by ensuring that the required number of ratings (one-third for this case) are retrieved only after the defined depth is *queried*. We can see that a consumer needs to wait a little more under this model, due to the limited availability of required ratings.

In the experiments above, we have used very basic versions for each dissemination protocol. Although the performance times are acceptable, we believe that with protocol refinements these times can be improved. We have seen that Publish-Subscribe Model provides best times followed by the Credibility-Based Model. The number of messages exchanged and hence the network load in these models is low as

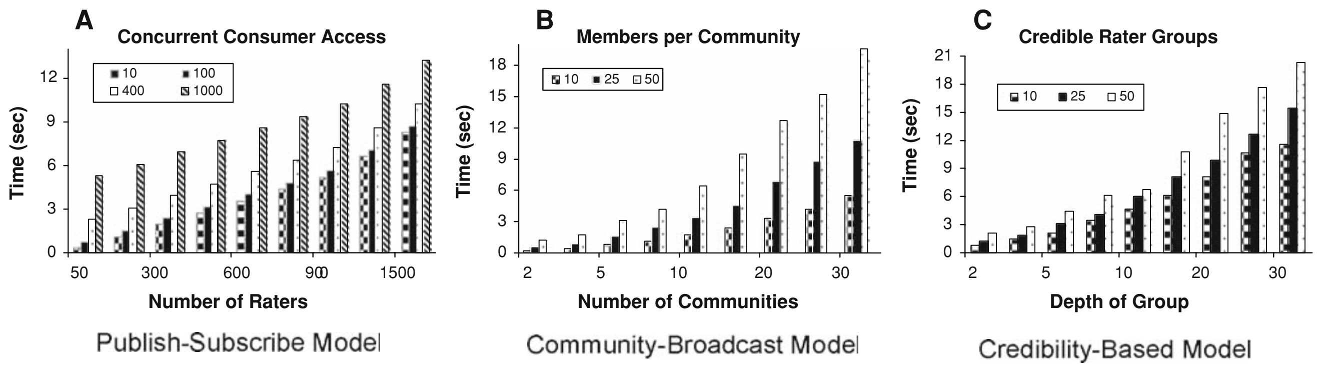

Fig. 16 RATEWeb performance with changing default values

compared to the Community-Based model. However, when the number of raters is low, we can see that the three model times are very comparable. The choice of the model (in regards to computation time and reputation accuracy) will thus depend on the domain, and type of interactions for which the services are deployed.

6 Related work

Reputation management involves several components, including modeling, data collection, data storage, communication, assessment, and reputation safeguards. Over the years, several research initiatives have worked on these problems. These efforts have not been limited to a single field. Varied disciplines including economics, computer science, marketing, politics, sociology, and psychology have studied reputation in several contexts [14]. In the recent past, these research activities have gained momentum. In computer science, reputation has been studied both in theoretical areas and practical applications. Theoretical areas where reputation has been studied include game theory, Bayesian networks, overlay networks and social networks to name a few. Theoretical literature that addressed reputation focused on proving properties of systems based on reputation. Major applications where reputation has been effectively used include e-business, peer-to-peer (P2P) networks, grid computing systems, multi-agent systems, Web search engines, and ad-hoc network routing. In the following, we give a brief overview of a few reputation management frameworks for P2P systems and Web services since these are closely related to our research.

PeerTrust [68] is a P2P reputation management framework used to quantify and compare the trustworthiness of peers. In PeerTrust, the authors have proposed to decouple feedback trust from service trust, which is similar to the approach undertaken in this paper. Similarly, it is argued that peers use a similarity measure to weigh opinions of those peers highly who have provided similar ratings for a common set of past partners. However, this may not be feasible for large P2P systems, where finding a statistically significant set of such past partners is likely to be difficult. Consequently, peers will often have to make selection choices for peers which have no common information in the system.

In [25], the *EigenTrust* system is presented, which computes and publishes a global reputation rating for each node in a network using an algorithm similar to Google's *PageRank* [48]. Each peer is associated with a global trust value that reflects the experiences of all the peers in the network with that peer. EigenTrust centers around the notion of transitive trust, where feedback trust and service trust are coupled together. Peers that are deemed honest in resource sharing are also considered credible sources of ratings information. This is in contrast with our approach and we feel this approach may not be accurate. Moreover, the proposed algorithm is complex and requires strong coordination between the peers. A major limitation of EigenTrust is that it assumes existence of pre-trusted peers in the network.

PowerTrust [71] is a "distributed version" of EigenTrust. It states that the relationship between users and feedbacks on eBay follow a Power-law distribution. It exploits the observation that most feedback comes from few "power" nodes to construct a robust and scalable trust modeling scheme. In PowerTrust, nodes rate each interaction and compute local trust values. These values are then aggregated to evaluate global trust through random walks in the system. Once power nodes are identified, these are used in a subsequent look-ahead random walk that is based on Markov chain to update the global trust values. Power nodes are used to assess the reputation of providers in a "system-wide absolute" manner. This is in contrast with our approach where each consumer maintains control over the aggregation of ratings to define a provider's reputation. Moreover, PowerTrust requires a structured overlay (for DHT), and the algorithms are dependent on this architecture. In contrast, service-oriented environments or the Web in general do not exhibit such structure.

PRIDE [16] is a P2P reputation framework that uses an elicitation-storage protocol for exchange of recommendations. The peers maintain a certificate authority which is responsible for the identity certificate of the peer. IP-Based Safeguard (IBS) is used to counter possible dishonesty since self-certification can lead to a peer generating a large number of identities for malicious reasons. Simple arithmetic average of recommendations received by a service provider is proposed to assess peer reputation. However, such an approach based solely on the sum of negative and positive ratings alone is vulnerable to unfair rating attacks, and hence may not be appropriate.

The XRep system proposed in [12] uses a combination of peer-based reputations and resource-based reputations to evaluate a peer's honesty. In this scheme, storage overheads are substantially high while incorporating resource-based reputations, as the number of resources is significantly more than the number of peers. Moreover, the experiments consider a Zipf (non-uniform) distribution of resources and peers. However, it may not be practical to consider a single resource to be widespread enough to have a sufficient number of ratings in the system. Similar to our approach, XRep uses cluster computing to weigh feedbacks and detect malicious parties. However, no formalized trust metric is discussed in the paper.

The P-Grid approach proposed in [1] assumes that most peers in the network are honest. The reputations in the system based solely on complaints. Though the method works well in the proposed scheme, it may not be robust. For instance, security concerns can arise if a peer ends up storing its own information. This is stated to be a rare case and redundancy is proposed to be employed to ensure data integrity. Moreover, since trust is only represented in binary values (1 and -1), the robustness is questionable. Either an agent will be completely trustworthy or untrustworthy. In our proposed system, the varying degrees of reputation solve this problem.

REGRET [54] is a reputation system that adopts a sociological approach for computing reputation in multi-agent societies in an e-commerce environment. Similar to our approach where the nature of the community effects the service's reputation, REGRET employs both individual and social components of social evaluations where the social dimension refers to reputation inherited by individuals from the groups they belong to. However, the proposed scheme requires a minimum number of interactions to make correct evaluations of reputation. It is likely that partners will not interact the minimum number of times to provide a reliable reputation value. Moreover, the problem of malicious raters is not studied.

In [24], a system of Reputation-agents (ˊf4R-agentsˊf6) that buy and sell reputation information on prospective partners is proposed. System participants choose R-agents on the basis of their personal experiences with the success rate of a particular R-agent and truthfulness is computed as statistical similarity. In other words, if the rating is the same as the last report about that agent, then it is deemed honest. Therefore, if a significant proportion of the agents are lying, then the definition of truth is inverted. This is similar to our approach. However, unlike our approach no checks on rater credibilities are placed to counter maliciousness.

In the networks domain, the CONFIDANT protocol is proposed in [6] where each node monitors its neighbors' behavior and maintains a reputation for each neighbor. This reputation is then propagated to the other nodes of the network. Nodes only accept feedback information from other nodes if those are close (within a threshold) to the current values held by the nodes. This method has two main limitations. First, this method does not consider the majority opinions when aggregating feedbacks and feedbacks that are different from a node's personal experience are rejected. This may not be true in general, since one single node's experience might not reflect the target node's behavior. Second, a Beta distribution is assumed for the node behavior.

Despite the abundance in reputation-related literature, little research has focused on the reputation of Web services. In [43], a distributed model for Web service reputation is presented. The model enables a service's clients to use their past interactions with that service to improve future decisions. It also enables services' clients to share their experience from past interactions with Web services. Agents are associated with each Web service, that act as proxies to collect information on and build a reputation of a Web service. The authors present an approach that provides a conceptual model for reputation that captures the semantics of attributes. The semantics includes characteristics, which describe how a given attribute contributes to the overall rating of a service provider and how its contribution decays over time. A similar reputation-based model using a node's first hand interaction experience is presented in [53]. The goal of the model is to increase/maintain QoS values in *selfish* overlay networks. The authors show that in presence of a reputation management system, an overlay network discourages *selfish* nodes. This increases the QoS guarantees in the network. The proposed model considers a node's first hand interaction experience and peer testimonials for deriving node reputations. In this regard, the reputation building process in [53] is similar to our approach. However, the proposed reputation model may not be completely robust and may not provide accurate results. First, the individual experience takes time to evolve over repeated interactions. Second, no distinction is made between the node's service credibility in satisfying consumer requests and its rating credibility. It may be the case that a node performs satisfactorily but does not provide authentic testimonials. We provide an extensive mechanism to overcome these and similar inadequacies.

In [40], the authors present an ontology model to aid in establishing trust in Web services. The trust model is refined in [41,42] and a trust model based on a shared conceptualization of quality of service (QoS) attributes is presented. The model shares the need for ontologies with our presented model. However, it lacks some important features

Table 3 Related work summary: in comparison with RATEWeb

System	Limitation	RATEWeb
PeerTrust [68]	• A common set of past providers is required "similarity" (for assessing rater credibility)	• Not required
EigenTrust [25]	• Feedback trust and service trust are coupled together • Assumes existence of pre-trusted peers	• Rater credibility (Feedback trust) is assessed separately from Provider Trust (service trust)
PowerTrust [71]	• Power nodes assigned higher credibility • Assumes a power-law distribution • Requires a structured overlay • System-wide reputation	• Credibility depends on rating honesty • No distribution assumed • Not required • User-centric reputation
PRIDE [16]	• IP-Based Safeguard (IBS) is used to counter possible dishonesty • Based on simple average of negative and positive ratings • Focused primarily on efficiency / performance (rather than functionality)	• Service identities are not "strict" (i.e. not tied to IPs) • Based on defined metrics • Focus on functionality in this paper (Performance study is on-going work)
XRep [12]	• No formalized trust metric is discussed • Data needs to be distributed according to a certain statistical distribution (Zipf)	• Formal metrics discussed • Services (and data) can be spread. No assumption is made
PGrid [1]	• Assumes that most peers in the network are honest • Representation of trust in binary values only	• Honest and dishonest services co-exist (in any number) • Reputation (hence trust) measured on a continuous scale from 0 to 1
REGRET [54]	• Requires a minimum number of interactions between providers and consumers to make correct evaluations of reputation • Raters dishonesty not discussed	• No minimum number of interactions required • Rater dishonesty assessment is an integral part of the approach
R-agents [24]	• If the rating is the same as the last report about that agent, then it is deemed honest	• Rater credibility assessment incorporates this approach and extends it
CONFIDANT [6]	• Each node monitors its neighbors' behavior and maintains a reputation for it • A Beta distribution is assumed for the node behavior	• Services cannot monitor their peers (normal Web-based interactions) • No distribution is assumed
QoS in Overlay Networks [53]	• Requires a minimum number of interactions between providers and consumers to make correct evaluations of reputation • Feedback trust and service trust are coupled together	• No minimum number of interactions required • Rater credibility (Feedback trust) is assessed separately from Provider Trust (service trust)
Web Services [41, 42]	• Human participation for feedback reporting • All agents that report reputation ratings are assumed to be trustworthy • Ratings are communicated to common agencies for aggregation which are expected to behave honestly	• No human intervention required • Raters consist of both honest and dishonest services • No central/common agency consulted for ratings aggregation (Each service performs this locally)

that are central to our proposed model. The reputation-based trust model in [42] lacks complete automation of feedback reporting. Human participation is necessary for rating Web services. Moreover, all agents that report reputation ratings are assumed to be trustworthy. Similarly, the common agencies to whom these ratings are communicated for sharing/aggregation are also expected to behave honestly. In our model, no such simplifying assumption is made. We calculate the reputation of a Web service based on the testimonies of both *trusted* and *malicious* raters. We provide an elaborate method to measure the credibilities of service raters. The credibility-based scheme allows us to assign more weights to the trustworthy testimonies as compared to untrustworthy ones. This feature was deemed as "future work" in [42]. Another feature that is absent in the previous models, but is present in ours is the incorporation of "local historical information" with the "assessed reputation view." Moreover, our model allows Web services to personalize their attribute preferences as in [42]. However our model also accepts single reputation values (that are to be calculated from attribute aggregations) if the preferences of the service requester are similar to those of the rater. This reduces the reputation computation time, which is a requirement in real-time situations for Web service selection and service query optimization.

In [18], a principal might trust an object if that object is trusted by a third party that is trusted by the given principal. This is similar to the notion of endorsement proposed in [39].

A key difference between the two approaches is that [18] captures policies for endorsement and delegation, whereas [39] seeks to capture service attributes and how they can be combined to support various policies. In [33], the authors present a framework for reputation-based service selection in semantic grids. The framework consists of a matchmaking service, a composer service, and a reputation manager service (RMS). Service consumers provide their ratings of services to the RMS. The RMS computes the reputation of a service based on the ratings for that service received from different users. In [17], the authors propose a Web service discovery method that considers both the functionality and the behavior of the Web services, while providing a scalable reputation model for ranking the discovering services. The method operates over a peer-to-peer system, thus avoiding the inherent problems of centralized systems such as scalability, single point of failure and high maintenance cost. Table 3 shows some limitations of the previous works described above and the advantage RATEWeb provides over those works.

7 Conclusion and future work

We have presented RATEWeb, a reputation management framework to establish trust among Web services. The framework is extensible and can be deployed in other contexts. We focused on a Peer-to-Peer (P2P) Web service environment where Web services can act as both consumers (i.e., requesters) and providers of services with out the need of a *trusted third party*. The details of the reputation assessment component and preliminary algorithmic details of the proposed framework are presented. We have also conducted extensive experiments to verify the presented framework. Results from the experiments exhibit strong evidence that the proposed RATEWeb approach provides a fairly accurate assessment of provider reputations.

In the future, we aim to extend the performance study by comparing dissemination models (while incorporating others) with each other in a more detailed manner. We will also extend and incorporate techniques to address methods for reputation bootstrapping [37], automatic rating of services, as well as extend the number of operations per service. Another important research direction is defining a reputation model for *composed* Web services.

Acknowledgments This work was supported by the US National Science Foundation's Cyber Trust Program under grant 0627469.

References

1. Aberer, K., Despotovic, Z.: Managing trust in a peer-2-peer information system. In: *CIKM*, pp. 310–317 (2001)
2. Al-Masri, E., Mahmoud, Q.H.: Discovering the best web service. In: 16th International Conference on World Wide Web (WWW), pp. 1257–1258 (2007)
3. Ali, A.S., Rana, O.F., Al-Ali, R., Walker, D.W.: Uddie: an extended registry for web services. Saint-w, 00:85 (2003)
4. Berners-Lee, T., Hendler, J., Lassila, O.: The Semantic Web. Scientific American, New York (2001)
5. Bertino, E., Ferrari, E., Squicciarini, A.C.: Trust-X: a peer-to-peer framework for trust establishment. IEEE Trans. Knowl. Data Eng. **16**(7), 827–842 (2004)
6. Buchegger, S., Le Boudec, J.-Y.: Performance analysis of the CONFIDANT protocol. In: Proc. of the 3rd ACM Intl. Symposium on Mobile Ad Hoc Networking and Computing, pp. 226–236, June 9–11 (2002)
7. Bouguettaya, A., Ouzzani, M., Medjahed, B., Cameron, J.: Managing government databases. Computer **34**(2), 56–64 (2001)
8. Buchegger, S., Le Boudec, J.-Y.: A robust reputation system for p2p and mobile ad-hoc networks. In: Proceedings of the Second Workshop on the Economics of Peer-to-Peer Systems (2004)
9. Buskens, V.: Social networks and the effect of reputation on cooperation. In: Proc. of the 6th Intl. Conf. on Social Dilemmas (1998)
10. Casare, S., Sichman, J.: Towards a functional ontology of reputation. In: AAMAS '05: Proceedings of the Fourth International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 505–511. ACM Press, New York (2005)
11. Casati, F.: Open issues and opportunities in web services modeling, development, and management. In: Coordination, Pisa, Italy, February (2004)
12. Damiani, E., De Capitani di Vimercati, S., Paraboschi, S.: Pierangela Samarati, Violante, F.: A reputation-based approach for choosing reliable resources in peer-to-peer networks. In: ACM Conference on Computer and Communications Security, pp. 207–216 (2002)
13. Delgado, J., Ishii, N.: Memory-based weighted-majority prediction for recommender systems. In: ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation (1999)
14. Dellarocas, C.: The digitalization of word-of-mouth: promise and challeges of online feedback mechanisms. Manage. Sci. **49**(10), 1407–1424 (2003)
15. Demirkan, H., Goul, M., Brown, G.W.: Towards the service oriented enterprise. In: 40th Annual Hawaii International Conference on System Sciences (HICSS'07), p. 62 (2007)
16. Dewan, P., Dasgupta, P.: Pride: peer-to-peer reputation infrastructure for decentralized environments. In: WWW (Alternate Track Papers & Posters), pp. 480–481 (2004)
17. Emekci, F., Sahin, O.D., Agrawal, D., El-Abbadi, A.: A peer-to-peer framework for web service discovery with ranking. In: Intl. Conf. on Web Services (ICWS) (2004)
18. Finin, T.W., Joshi, A.: Agents, trust, and information access on the semantic web. ACM SIGMOD Rec. **31**(4), 30–35 (2002)
19. Garcia, D., Toledo, M.: A web service architecture providing qos management. In: LA-WEB '06: Proceedings of the Fourth Latin American Web Congress (LA-WEB'06), pp. 189–198. IEEE Computer Society, Washington (2006)
20. Giuliano, G.: ARGOS: Dynamic Composition of Web Services for Goods Movement Analysis and Planning. Technical report, University of Southern California (2003)
21. Houser, D., Wooders, J.: Reputation in auctions: theory, and evidence from eBay. J. Econ. Manage. Strategy **15**, 353–369 (2005)
22. Huynh, T.D., Jennings, N.R., Shadbolt, N.R.: Certified reputation: how an agent can trust a stranger. In: AAMAS '06: Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems, pp. 1217–1224. ACM Press, New York (2006)
23. Josang, A., Ismail, R., Boyd, C.: A survey of trust and reputation systems for online service provision. Decis. Support Syst. **43**(2), 618–644 (2007)

24. Jurca, R., Faltings, B.: An incentive compatible reputation mechanism. In: Proc. of the 2003 IEEE Intl. Conf. on E-Commerce, pp. 285–292 (2003)
25. Kamvar, S.D., Schlosser, M.T., Garcia-Molina, H.: The eigentrust algorithm for reputation management in p2p networks. In: Proceedings of the Twelfth International World Wide Web Conference (WWW) (2003)
26. Kesler, C.: Experimental games for the design of reputation management systems. IBM Syst. J. **42**(3), 498–506 (2003)
27. Lam, S.K., Riedl, J.: Shilling recommender systems for fun and profit. In: Proc. of the 13th International World Wide Web Conference (WWW), pp. 393–402, New York, NY, USA (2004)
28. Lamanna, D., Skene, J., Emmerich, W.: Slang: A language for defining service level agreements. In: Proc. of the 9th IEEE Workshop on Future Trends in Distributed Computing Systems—FTDCS 2003, pp. 100–106. IEEE-Computer Society, New York (2003)
29. Li, Z., Wang, S.: The foundation of e-commerce: social reputation system—a comparison between american and china. In: ICEC '05: Proceedings of the 7th International Conference on Electronic Commerce, pp. 230–232. ACM Press, New York (2005)
30. Liu, Y., Ngu, A., Zheng, L.: Qos computation and policing in dynamic web service selection. In: WWW Alt. '04: Proceedings of the 13th International World Wide Web Conference on Alternate Track Papers & Posters, pp. 66–73. ACM Press, New York (2004)
31. Ma, W., Tosic, V., Esfandiari, B., Pagurek, B.: Extending apache axis for monitoring of web service offerings. In: BSN '05: Proceedings of the IEEE EEE05 International Workshop on Business Services Networks, pp. 7–14. IEEE Press, Piscataway (2005)
32. MacQueen, J.B.: Some methods for classification and analysis of multivariate observations. In: Proceedings of 5-th Berkeley Symposium on Mathematical Statistics and Probability, pp. 281–297. University of California Press, Berkeley (1967)
33. Majithia, S., Ali, A.S., Rana, O.F., Walker, D.W.: Reputation-based semantic service discovery. In: Proc. of the 13th IEEE International Workshops on Enabling Technologies: Infrastructures for Collaborative Enterprises (WETICE), pp. 297–302 (2004)
34. Malaga, R.: Web-based reputation management systems: problems and suggested solutions. Electron. Commer. Res. **1**(1), 403–417 (2001)
35. Malik, Z., Bouguettaya, A.: Evaluating rater credibility for reputation assessment of web services. In: 8th International Conference on Web Information Systems Engineering (WISE 07), December (2007)
36. Malik, Z., Bouguettaya, A.: Rater credibility assessment in web services interactions. In: World Wide Web Journal (WWWJ), pp. 1–23 (2008). http://dx.doi.org/10.1007/s11280-008-0056-y
37. Malik, Z., Bouguettaya, A.: Reputation bootstrapping for trust establishment among web services. IEEE Internet Comput. **13**(1), 40–47 (2009)
38. Mani, A., Nagarajan, A.: Understanding quality of service for web services (2002). http://www-128.ibm.com/developerworks/library/ws-quality.html
39. Maximilien, E.M., Singh, M.P.: Reputation and endorsement for web services. SIGecom Exch. **3**(1), 24–31 (2002)
40. Maximilien, E.M., Singh, M.P.: A framework and ontology for dynamic web services selection. IEEE Internet Comput. **8**(5), 84–93 (2004)
41. Maximilien, E.M., Singh, M.P.: Toward autonomic web services trust and selection. In: ICSOC 2004: Proceedings of 2nd International Conference on Service Oriented Computing, November (2004)
42. Maximilien, E.M., Singh, M.P.: Agent-based trust model involving multiple qualities. In: AAMAS 2005: Proceedings of 4th International Autonomous Agents and Multi Agent Systems, July (2005)
43. Maximillien, E.M., Singh, M.P.: Conceptual model of web service reputation. SIGMOD Record **31**(4), 36–41 (2002)
44. Medjahed, B., Bouguettaya, A.: Customized delivery of e-government web services. IEEE Intell. Syst. **20**(6), 77–84 (2005)
45. Medjahed, B., Bouguettaya, A., Elmagarmid, A.: Composing web services on the semantic web. VLDB J. **12**(4), 333–357 (2003)
46. Medjahed, B., Rezgui, A., Bouguettaya, A., Ouzzani, M.: Infrastructure for e-government web services. IEEE Internet Comput. **7**(1), 58–65 (2003)
47. Medjahed, B.: Semantic Web Enabled Composition of Web Services. Ph.D. thesis, Department of Computer Science, Virginia Tech, January (2004)
48. Page, L., Brin, S., Motwani, R., Winograd, T.: The PageRank Citation Ranking: Bringing Order to the Web. Technical report, Stanford Digital Library Technologies Project (1998)
49. Papazoglou, M.P., Georgakopoulos, D.: Serive-oriented computing. Commun. ACM **46**(10), 25–65 (2003)
50. Park, S., Liu, L., Pu, C., Srivatsa, M., Zhang, J.: Resilient trust management for web service integration. In: ICWS '05: Proceedings of the IEEE International Conference on Web Services (ICWS'05), pp. 499–506. IEEE Computer Society, Washington (2005)
51. Ran, S.: A model for web services discovery with qos. SIGecom Exch. **4**(1), 1–10 (2003)
52. Resnick, P., Zeckhauser, R.: Trust among strangers in internet transactions: empirical analysis of eBay's reputation system. Adv. Appl. Microecon. **11**, 127–157 (2002)
53. Rocha, B.G., Almeida, V., Guedes, D.: Increasing qos in selfish overlay networks. IEEE Internet Comput. **10**(3), 24–31 (2006)
54. Sabater, J., Sierra, C.: Reputation and social network analysis in multi-agent systems. In: Proc. of the first Intl. Joint Conf. on Autonomous Agents and Multiagent Systems, pp. 475–482, Bologna, Italy (2003)
55. Sabou, M., Wroe, C., Goble, C., Mishne, G.: Learning domain ontologies for web service descriptions: an experiment in bioinformatics. In: 14th International World Wide Web Conference (WWW) (2005)
56. Skogsrud, H., Benatallah, B., Casati, F.: Trust-serv: a lightweight trust negotiation service. In: VLDB Demo, Toronto, Canada, August (2004)
57. Sonnek, J.D., Weissman, J.B.: A quantitative comparison of reputation systems in the grid. In: The 6th IEEE/ACM International Workshop on Grid Computing, pp. 242–249, November (2005)
58. Sundaresan, N.: Online trust and reputation systems. In: EC '07: Proceedings of the 8th ACM Conference on Electronic Commerce, pp. 366–367. ACM Press, New York (2007)
59. Tennenholtz, M.: Reputation systems: an axiomatic approach. In: AUAI '04: Proceedings of the 20th Conference on Uncertainty in Artificial Intelligence, pp. 544–551. AUAI Press, Arlington (2004)
60. Tsai, W.T.: Service-oriented system engineering: a new paradigm. In: IEEE International Workshop on Service-Oriented System Engineering (SOSE 05), pp. 3–6, October (2005)
61. Udupi, Y.B., Singh, M.P.: Information sharing among autonomous agents in referral networks systems. In: 6th International Workshop on Agents and Peer-to-Peer Computing, May (2007)
62. Vogel, A., Kerherve, B., Bochmann, G., Gecsei, J.: Distributed multimedia and qos: a survey. IEEE Multimed. **2**(1), 10–18 (1995)
63. Vu, L.-H., Hauswirth, M., Aberer, K.: QoS-based service selection and ranking with trust and reputation management. In: 13th International Conference on Cooperative Information Systems (CoopIS 2005), October 31–November 4 (2005)
64. Walsh, K., Sirer, E.G.: Fighting peer-to-peer spam and decoys with object reputation. In: P2PECON '05: Proceedings of the 2005 ACM SIGCOMM Workshop on Economics of Peer-to-peer Systems, pp. 138–143. ACM Press, New York (2005)

65. Weng, J., Miao, C., Goh, A.: Protecting online rating systems from unfair ratings. Trust, Privacy and Security in Digital Business. Lecture Notes in Computer Science
66. Whitby, A., Josang, A., Indulska, J.: Filtering out unfair ratings in bayesian reputation systems. Icfain J. Manage. Res. **4**(2), 48–64 (2005)
67. Wu, W., Doan, A., Yu, C., Meng, W.: Bootstrapping domain ontology for semantic web services from source web sites. In: VLDB Workshop on Technologies for E-Services (2005)
68. Xiong, L., Liu, L.: PeerTrust: supporting reputation-based trust for peer-to-peer electronic communities. IEEE Trans. Knowl. Data Eng. (TKDE) **16**(7), 843–857 (2004)
69. Yang, J., Wang, L., Zhang, S., Sui, X., Zhang, N., Xu, Z.: Building domain ontology based on web data and generic ontology. In IEEE/ACM International Conference on Web Intelligence (WI 2004), pp. 686–689, September (2004)
70. Yu, Q., Liu, X., Bouguettaya, A., Medjahed, B.: Deploying and managing web services: issues, solutions, and directions. VLDB J. **17**(3) 537–572 (2008)
71. Zhou, R., Hwang, K.: Powertrust: a robust and scalable reputation system for trusted peer-to-peer computing. IEEE Trans. Parallel Distributed Syst. **18**(4), 460–473 (2007)