Analysis of Trust-Based Approaches for Web Service Selection

Dragoni, Nicola; Miotto, Nicola

Published in:
Proceedings of the 5th Nordic Workshop on Dependability and Security (NODES’11)

Publication date:
2011

Link back to DTU Orbit

Citation (APA):

General rights
Copyright and moral rights for the publications made accessible in the public portal are retained by the authors and/or other copyright owners and it is a condition of accessing publications that users recognise and abide by the legal requirements associated with these rights.

- Users may download and print one copy of any publication from the public portal for the purpose of private study or research.
- You may not further distribute the material or use it for any profit-making activity or commercial gain
- You may freely distribute the URL identifying the publication in the public portal

If you believe that this document breaches copyright please contact us providing details, and we will remove access to the work immediately and investigate your claim.
Analysis of Trust-Based Approaches for Web Service Selection

Nicola Dragoni, Nicola Miotto

Department of Informatics and Mathematical Modelling
Technical University of Denmark, 2800 Kongens Lyngby, Denmark
ndra@imm.dtu.dk

Abstract. The basic tenet of Service-Oriented Computing (SOC) is the possibility of building distributed applications on the Web by using Web services as fundamental building blocks. The proliferation of such services is considered the second wave of evolution in the Internet age, moving the Web from a collection of pages to a collections of services. Consensus is growing that this Web service revolution wont eventuate until we resolve trust-related issues. Indeed, the intrinsic openness of the SOC vision makes crucial to locate useful services and recognize them as trustworthy. In this paper we review the field of trust-based Web service selection, providing a structured classification of current approaches and highlighting the main limitations of each class and of the overall field.

1 Introduction

Service Oriented Computing is an emerging paradigm for distributed computing, where Web Services (WS) represent the bricks of a Service Oriented Architecture. Brought to its full potential, this vision could allow software developers to take advantage of agents to automatically discover and and compose Web Services over the internet to build a distributed system. In the past years, there have been many issues discussed about web services, regarding their implementation, their founding principles and on. But there is still a concern that did not get much attention so far and would need to be thoroughly investigated: how can a service be trusted? What is the right choice when it comes to decide the best service among plenty of similar ones? In this paper we are going show how it has been tried to answer this question, providing a survey about the state of the art of the web service trust provisioning.

1.1 Centralized vs Distributed

Trust provisioning systems can be built mainly around 2 architectures (apart from TN): centralized or distributed. Even if each single approach applies different rules for the entity-to-entity communication or uses different algorithms to compute the trust, the architecture-dependent behaviour can be generalized as follow:

Definition 1 - Distributed trust provisioning approaches: the trust scores of WSs / service providers are computed/derived after having directly communicated with other peers in the system.

Limitations: in the context of SOA is not really possible to generalize the shortcomings deriving from a distributed architecture, because they are tightly connected to the specific approach. Anyway, in general, a common drawback is to setup effort.

Definition 2 - Centralized trust provisioning approaches: the trust scores of WSs /
service providers are provided by a central authority with the responsibility of computing/collecting them.

**Limitations:** Central authorities are a single point of failure and thus can exist only under rigidly constructed and administered computational environments, in particular considering the capacity demand of a SOA environment. Another technical limitation resides in the possible alteration of the ratings (collusion or retaliation). Moreover, a centralized trust authority can never be a good enough recommender for everyone: different entities should be allowed to make up their own mind[1].

Further advantages and drawbacks will be discussed in the following sections according to each approach.

### 1.2 Sources of trust

The studies we are going to analyze throughout this paper adopt different metrics to evaluate the trustworthiness of a WS. The most part of those metrics boils down to the QoS metrics for WSs identified by W3C, as shown in figure 1:

![Fig. 1. QoS metrics according to the W3C Consortium](image_url)

### 1.3 Real World scenario: Virtual Tourism Agency

Alice is a software developer for a tourism company. She’s asked to develop a Virtual Tourism Agency (VTA), a service helping the users throughout the steps to plan a trip; she decides to break down the system to its smaller capabilities: flight booking, accommodation booking, bus ticket purchase/car rent, payment. Since there are many WSs providing for the identified features, she has to choose the right ones and compose them in a step by step procedure; but, which service is the right one? Alice didn’t use any of the available services in her career, so she picks them up just relying on few descriptions.
and her common sense; after one month, the company providing for the flight booking WS has to temporary shut down the service because of overloading problems. Consequently, the VTA results as to be taken down as well.

As the example tries to stress, selecting the right service does not include only the problem of discovering services on the basis of what a service can do (functional properties), but also how well a service can do (nonfunctional properties), evaluated according to some non-functional QoS\(^1\) metrics. The QoS value may determine the trust Alice has towards a service.

2 Suggested approaches

There have been many studies in literature addressing the problem of automatic trust provisioning, not necessarily directly targeting the WS domain. The purpose of this document is to classify the most relevant ones according to their rationale, in order to outline their advantages and limitations in a SOA environment.

The different studies can be grouped according to the following diagram [2]:

![Classification of Trust Approaches for Online Service Provision](image)

**Fig. 2.** Classification of Trust Approaches for Online Service Provision.

In the following sections we will discuss the suggested solutions and their limitations, providing an example scenario each time.

2.1 Direct Experience

This class of approaches is based on presumptions drawn from the service consumer’s own direct experience with the target service [3]. The rationale is that the trust can be build upon some quality parameters that depend on the service behaviour in the course of time. This means that the service can be trusted if the consumer past experience with that service, i.e. *the knowledge gained after having a transaction with it*[3], results complying to his expectations and requirements. Jonker and Jan Treur present an analysis of models for the dynamics of trust based on experiences [4]. They investigate the basing principles governing the evolution of the trust an agents has towards a service. In [5] the authors describe a layered framework conceived to manage trustworthiness through seven levels. As the authors point out *the quality of the model to be built is fully dependent on the experience of the practitioners*. In other words, how to trust a service

\(^{1}\) Quality Of Service
when no past experience is available? This is the main drawback of the trust-by-direct-experience approaches.

**Definition 3 - Trust by Direct Experience:** a service consumer trusts a service because of his good past experience with the service.

**Limitations:** This approach is not suitable for large open systems where anyone can publish its (malicious) code, since it does not allow to trust a service before its execution. Moreover, whenever an *unconditional distrust* approach is used, brand new services may be not considered even if conforming to the needs.

**Scenario 1:** Alice has no past experience with any flight booking WS. For each discovered WS, with no evidence of its trustworthiness, she is forced to either unconditionally trust or distrust it. In the first case Alice has to accept not only the WS inherent “risk of prior performance” (i.e., to pay for services and goods before receiving them) but also the “risk of blind (i.e., untrusted) execution”. In the second case she is going to reject a service that might have been compliant to her trust policies.

### 2.2 Trusted Third-Party Approaches

TTP approaches are based on the idea that the service consumer can rely on a third-party in order to obtain a trust value of a given service. *Third-party* may refer either to a trusted central authority or members of a community. The underlying assumption of these approaches is that consumers must trust the third party they decide to consult. We distinguish among two types of approaches: social and matchmaking approaches. In both of them the final decision is based on the assessments provided by the TTP. The difference lies in how the assessments are computed.

**Social (Indirect Experience)** The trust evaluation towards a WS is forged by a cooperating community whose members have directly or indirectly interacted with such WS. In order to be effective, each community member has to continuously review the services (and the service providers) it’s using. The global evaluation is not necessarily calculated by the community members themselves, but might be the result of a centralized data mining applied on member-supplied informations. In literature we can find three different social-based approaches: reputation, recommendation and referral.

**-Reputation:** The definition of the term reputation from The English Oxford Dictionary is: “a widespread belief that someone or something has a particular characteristic.”

In a SOA context, the rationale is that a WS is trustworthy as long as the community has a good opinion about it. The reputation system is responsible to collect ratings about users, services and service providers from members in the community.

The global opinion can be modeled either around the QoS parameters described above or depending to what degree WSs abide by the contract. However, the parameters used to rate a service, user or service provider are not influencing the rationale of a TTP reputation system: an individual’s subjective trust on a service is derived from the reputation of that service or, in other words, from the direct experience of someone else.

**Definition 4 - Trust by Reputation:** a service consumer trusts a service because of his good reputation.

A major distinction between different reputation systems is outlined by the base architecture: centralized or distributed [6][7]. Most of the studies suggests centralized
approaches\cite{8}, typical of e-commerce web sites\cite{2}. In those systems a central authority is responsible to collect all the ratings from other members in the community (e.g. QoS data from WS consumers, in our case) who have had direct experience with a specific service or provider. The authority uses these ratings to derive a reputation score for the service and makes it publicly available to future, potential consumers. Regarding distributed reputation systems, two notable examples are EigenTrust \cite{9} and the PeerTrust \cite{10}. Each member of the community (be it an agent or a human) records its own opinion about a service to make it available to the others. A reputation grade is a function of all the trust ratings (if there are any) obtainable by all the possible members the agent can reach.

**Limitation A**: the effectiveness of any reputation system lies on the number of members in a community and on their behavior. The fewer the members in a reputation system, the more inadequate the ratings provided by the systems. This issue envelopes the community-bootstrap problem:

**Definition 5 - Community bootstrap issue**: a community-dependent system is unlikely to provide good quality results as long as the community is small or not really active.

**Limitation B**: another shortcoming is that trust relies on past information from other members of the community. A natural problem arises in case of new services. For example, when a service initially registers for business, no other consumer has interacted with it and consequently no information exists about its past behaviour and questions about its trustworthiness are left unanswered. This can be defined as the *new WS ramp-up* issue:

**Definition 6 - New WS ramp-up issue**: a new web service needs time before being adequately evaluated.

**Scenario 2**: Alice needs a trust score for a just discovered flight booking service. She sets up an agent to query all the neighbour agents for a reputation grade. The agents (based on theirs and their neighbours past experience) return very low scores to Alice’s agent. The WS is then discarded, but maybe either the community or the WS are new born and it comes difficult to provide a useful evaluation. Thus, Alice still doesn’t can’t adequately evaluate the WS trustworthiness.

Now, a question for this and other social approaches arises: how the community should repute a new service? There have been many studies addressing this issue. In the Sporas system suggested by Giorgios Zacharia et al., new users start with the minimum reputation value\cite{11}. The authors of the Dirichlet algorithm\cite{7} (conceived for P2P sharing network) state that *it is possible to track the average reputation score of the whole community, and this can be used to set the base rate for new agents*\cite{7}.

**Limitation**: in general, when the starting reputation is low, the new WS is underestimated. Whenever a new WS receives an initial reputation score higher than the minimum, this can be exploited by malicious users by continuously subscribing and unsubscribing to the system in order to keep having a “non zero” reputation value.

**Scenario 3**: Alice’s software agent starts discovering new flight booking WSs on behalf of Alice. It queries a TTP that states that the chosen service has a non zero grade. Alice still doesn’t know whether the service is trustworthy: it might belong to a malicious provider just subscribed to the community.

\textsuperscript{2} e.g. http://www.ebay.com
Recommendation: Recommendation systems [12][1] aim at making a prediction of a consumer’s needs of interests. In its common formulation [13], the recommendation problem is reduced to the problem of estimating ratings for the items (such as services) that have not been seen by a consumer. Intuitively, this estimation is usually based on the ratings given by this consumer to other items or on the ratings that similar users provided for the targeted items. Once it is possible to estimate ratings for the yet unrated items, then the system can recommend to the user the items with the highest estimated ratings.

Definition 7 - Trust by Recommendation: a service consumer trusts a service because of some recommendations got from a trusted authority.

In general, recommendation-based systems work as good as wide and rich the knowledge of the system is. In other words, it is necessary to know both the community and the user requesting the service in order to produce reasonable evaluations. Recommender systems can be classified into five categories according to [2][14][12]: content-based filtering, the only static approach among the five ones, items are selected according to their content. But this approach is very primitive and would be a step backward from current WSs standards (which involve formal structured description of services); the most widely used one in e-commerce sites like Amazon [http://www.amazon.com] is the Collaborative Filtering (CF): the consumer is recommended items that people with similar tastes and preferences liked in the past. In CF, the implicit assumption is that different people have different tastes. Note that this represents the key difference with respect to reputation systems [15]. Items recommended to the user are then the ones other users with similar tastes (neighbours) liked [16]. The following three approaches are utility-based [17], demographic [18][19] and knowledge-based [20][21] recommenders, where basically, as for CF, by means of different classification and data mining algorithms, they infer the relationship between user need/profiles and items in the community (e.g. in our case they may be WSs and other community members). The last group is identified by the hybrid recommenders, whose rationale is to combine two or more recommendation techniques (usually with CF) to gain better performance with fewer of the drawbacks of any individual one [14].

Limitations: those typologies of recommendation system share one main weakness: the system needs many information about the users in order to provide useful evaluation. This can be achieved by either asking the users to disclose maybe sensitive information (as we have seen this is not suitable in a SOA environment) or by mining them out of the interaction of the users with the system, that would require a long time. Moreover, the well known issues of the social systems are still present, such like community boot-strap, new user and service ramp-up (somehow mitigated in the hybrid recommender). Finally, the recommendation systems are conceptually centralized (see section 1.1).

Definition 8 - New User ramp-up issue: a new user needs to interact with the trust provisioning system in order to receive good quality results.

Scenario 4: Alice finds a WS recommender service that requires to input some informations regarding the company past experience with other web services in order to be used. Alice’s company does not want to disclose this information to a central system. A second recommender service does not demand any pre-use information, but it requires
Alice’s agent to interact with the community for a period before being able to provide a recommendation and so Alice can’t use it.

**Referral:** Referrals [22][23] have been proposed as a decentralized approach based on online communities and software agents technologies. An online community is a set of interacting members (or principals in the jargon) representing people, businesses or other organizations. The members of a community provide services as well as referrals for services to each other. Referrals may be provided proactively or in response to requests. Members are assisted by software agents to help them manage their interactions[22]. Referrals are based on a representation of how much the other available parties can be trusted. Agents are responsible to build and manage these representations taking into account the previous experiences of their members and communicate with each others. Participating on behalf of different members, agents appear as autonomous and heterogeneous. Moreover, agents organize themselves into communities and agents in the same community are called neighbours. A key difference from the recommendation systems is that in referral systems the participants reveal their ratings to those whom they trust, so the ratings would be more likely to be honest.

**Definition 9 - Trust by Referrals:** A service consumer trusts a service because of some referrals obtained from trusted software agents.

**Limitations:** Referral systems address some limitations of reputation and recommendation systems (such as, their centralized nature) but still rely on the judgements of the members of a community and new WS are difficult to start-up. Moreover some technical practical issues, such as agents and members registration and communication as well as referrals representation, are left unanswered in the literature, making the impression of a still immature (or at least just academic) approach.

**Scenario 5:** refer to scenario 1 and 2.

**Matchmaking** These approaches are based on a component called “matchmaker” responsible to match a user’s request and trust preferences with available online service descriptions. If some matches are found than the results are sent back to the user.

A centralized trust-based matchmaking methodology has been proposed by Galizia et al. in [24]. Differently to other approaches, they embodied the WS selection problem in a classification problem: given a set of user and WS policies and established a classification criterion, the goal is to identify a class of WSs matching with trust policies of involved users. In other words, WSs are classified according to the specific user as well as trust policies.

**Definition 10 - Trust by Matchmaking:** A service consumer trusts a service because a trusted (central/distributed) matchmaker states that the service’s policy matches the consumer’s request.

**Limitation A:** matchmakers suffer of all the drawbacks inherited by the centralized architecture (see 1.1). Moreover, both consumers and providers has to register to the matchmaker in order to use it (far from the SOC vision). Finally, it is not realistic to ask the providers to disclose all teir (maybe sensitive) policies to a central authority.

Olmedilla et al. [25] replaces the centralized matchmaker and registry with a Peer-to-Peer network, distributing the matchmaking process to the service providers. A similar approach has been proposed by Olmedilla et al. in [25]. The main difference with respect to [24] lies in the underlying registry and matchmaking architecture, which is
based on a Peer-to-Peer network. Whenever a new service provider wants to offer its services, it must just join such network. On the client side, a user looking for a service must send a query along with his policies to a reasoning agent he trusts. The agent distributes the query to the peers on the network and each one of them applies a matching algorithm. Whenever a peer has matches, it sends them back to the reasoning agent which joins the results and presents them to the user. This way servers can keep policies locally and private. However the problem is moved from trusting a service or a service provider to the one of finding such a trusted reasoning agent.

**Scenario 6:** Alice, looking for a distributed matchmaking service, has to face the problem of selecting and trusting a reasoning agent. But where can she locate such computational entities? And which agents should she trust? On which basis? Current distributed matchmaking technologies do not answer, leaving Alice in the same (vulnerable) situation.

**Limitation B:** this family of approaches seems to solve the community-bootstrap problem. The trustworthiness score is evaluated matching the user trust requirements directly against the WS provided trust guarantees. However this approach exposes another issue: it wouldn’t be difficult for a malicious (or distracted) user to craft a WS description so to pretend to be a trustworthy WS. In this case a “watching” community would turn useful.

**Scenario 7:** Alice’s submitted the company’s trust profile to either a distributed matchmaker. Now a software agent is instructed to trust WSs providing providing AES encryption algorithm and capable of handling 20 simultaneous connections. The matchmaker returns to the agent the matching WSs and Alice selects one. During the pre-easter week (when many people use the VTA system), the system starts having many faults. Alice realizes that the flight booking WS cannot handle more than 10 connections simultaneously and the VTA system has to be temporary taken down.

### 2.3 Hybrid

Hybrid approaches for trust-based online service selection are based on a combination of well known trust methodologies, improving the quality of the assessments.

**Socio-Cognitive** These approaches are mostly based on the works of Falcone and Castelfranchi [26][27][28]. Influenced by the Artificial Intelligence (AI) field and especially by the Multi-Agent System (MAS) paradigm, they treat trust as an agent’s mental state. The agent supports beliefs from which is possible to derive a degree of trust. As pointed out in [3], beliefs can be seen as the answers to the question “What do we have in mind when we trust a service?”. According to these beliefs the agent can articulate assumptions and expectations about a specific service.

**Definition 11 - Socio-Cognitive Trust:** The degree of trust is a function of the subjective certainty of the pertinent beliefs. Therefore, A service consumer trusts a service because of some of its subjective beliefs.

The trust level is a function of such subjective beliefs. A key question therefore arises: how are such beliefs obtained? That is, from which sources? The answer to the above question differentiates the various proposals in literature. 2 of the most common sources of belief are the ones already discussed in the previous sections, i.e. direct experience (2.1) and reputation (2.2). Further on there are categorization (the process of grouping
things based on prototypes) and reasoning (the act of using reason to derive a conclusion from certain premises). For instance, in [4][29] the authors propose models in which they consider the direct interaction or reputation as sources. In [28] sources are categorization and reasoning. In [3] Ali et al. restrict sources to direct experience and reputation.

**Limitations:** a first weakness of the approach lies in the fact that it is based on beliefs obtained by means of the well known (and problematic) methodologies for trust. Another major limitation lies at the implementation level. To fully realize this approach, some sort of BDI agents [30] is needed. Indeed, as Falcone et al. remarks in their paper [26] only a cognitive agent can “trust” another agent. We mean: only an agent endowed with goals and beliefs. This requirement seems too strong when applied to open and large service-based systems, since it is not reasonable to assume that every agent will be conformed to the BDI model (which, a part from the modeling of trust, requires specific architectures to support the reasoning on beliefs and goals).

**Scenario 8:** Alice decides to delegate the trustworthy flight booking WS discovery to a BDI agent. Considering the openness of the system Alice is dealing with she realizes that her agent won’t be capable of communicating with many other agents of the same kind. The selected WSs are then narrowed to a small amount compared to the total. The chosen WS is thus going to be far away from the best available choice.

**Trust & Reputation** Studies such as [31][32][33] propose methods for assessing the quality of online services by combining trust and reputation techniques in a single integrated framework. For instance, [33] shows how (Bayesian) reputation systems can be combined with trust modeling based on subjective logic [34]. [31] describes a report-driven framework. WSs have their QoS profile computed by means of reports provided by both providers and consumers. The profiles are used to generate a WS rank based on the consumer requirements and reputation. Then, in order to prevent “spammers” or distracted users from poisoning the system, the authors suggest, along with the reputation framework, a trust system capable of identifying liars.

**Definition 12 - Trust & Reputation based system:** A system providing for a trustworthiness score employing methodologies based on both reputation and trust, in order improve some weaknesses of the constituent methodologies.

**Limitations:** Although these approaches are remarkable, especially [33] where the integration results in a flexible framework for online trust management, they still suffer the main limitations of their constituent methodologies. For instance, both approaches inherit one of the main weaknesses of some reputation systems (social and centralized) (section 2.2). The authors of [33] propose a bootstrapping method consisting of creating trusted reports for the most important WSs by means of trusted monitoring agents. However, with this approach there would be the problem of selecting the most important WSs.

**Scenario 9:** refer to scenario 1 and 2.

**Direct Experience & Reputation** [35][36][37] propose a model where the trust in a service is computed as a rating of the level of performance of the service. This overall performance is not limited to the agent’s direct experience (or confidence, see section

---

4 Belief-Desire-Intention
2.2) but it is also based on the evaluations of the service by other agents in the system (in [35] called the “group experience”, i.e., what the other members of the group think about the agent being evaluated and his group). Thus, in these models trust can be seen as a rating built as a result from combining agent’s direct experience (with the service) along with the social reputation of the service provider.

**Definition 13 - Trust by Direct Experience & Reputation:** The trust towards a service is evaluated by means of the user direct experience combined with the service reputation.

**Limitations:** again, the combination of two methodologies improve some weaknesses of one constituent model, but it does not provide a complete solution to the trustworthy online service selection problem. For instance, in [36] the authors combine confidence and reputation to address the situation where no previous experience of the service is available (main weakness of the direct experience method). But to do this they based their proposal on trust and reputation mechanisms to infer expectations of future providers’ behavior from past experiences in similar situations. This idea inherits the already discussed problems of trust and reputation mechanisms.

**Scenario 10:** Alice’s agent can’t establish a reliable trustworthiness score for certain flight booking WSs because there are no past interactions with them and, moreover, they seem to have joined the WS network too recently in order to have some useful reputation evaluations.

### 2.4 Automated Trust Negotiation

Automated Trust Negotiation (TN) [38] is an approach specifically targeted to allow agents to access sensitive data and services in open environments. Trust negotiation protocols are based on the iterative disclosure of digital credentials and requests for credentials between two unknown parties (strangers in TN jargon), with the goal of establishing sufficient mutual trust so that the parties can complete a transaction. Informally, digital credentials (credentials for short) refer to the online analogues of paper credentials (a drivers license, passport, or employee ID card, for example). Thus, a credential is a digitally signed assertion by a credential issuer about the credential owner. It is usually signed using the issuers private key and verified using the issuers public key [39]. To automate trust negotiation, each party must establish access control policies (policies for short) to protect its sensitive resources, including credentials and services, from inappropriate access. Each policy should specify the digital credentials strangers must present to access the protected resource. Policies can themselves be seen as sensitive resources. Considering that both the consumer and the provider can provide their own policies to gradually disclose, the point of view is not restricted to the service consumer only anymore (how the service consumer may trust a service): the goal now is to establish a mutual trust between service consumer and provider.

**Definition 14 - Credential-Based Trust (or Trust by Negotiation):** A service consumer and a service provider mutually trust each other because the access control policy of the requested service is compliant with the access control policy of the service consumer.

Note that the above definition does not state that a negotiation will always succeed if the parties’ policies are compliant. Indeed, the success of the negotiation depends on several factors. For instance, a negotiation could take different routes according to the negotiation strategies adopted by the parties [40]. The above definition just states that
if a trust negotiation succeeds establishing a mutual trust among two parties then this is because the two parties have compliant access control policies for the requested resource.

**Limitations**: trust negotiation principles and systems have been widely investigated in the last few years, both in different (still mainly academic) domains (like e-Business, e-Commerce, P2P systems and more recently in WSs [41]) and with respect to issues such as privacy, safety and efficiency. This effort is evident in the growing literature on TN related issues ([39][42][43][44][45] to mention only a few). However, several key issues have still to be addressed to bring TN to its full potential: to date, the proposed frameworks seems to have been studied in theoretical and academic fashion, still “unplugged” from the real nature of WS and SOA: first of all, many in the literature treated WSs (WS) as a set of independent single operations, while interacting with real world WSs involves generally a sequence of invocations of several of their operations [46]; then SOAP-based WSs are supposed to be consumed many times from the same costumer, since they are involved in a composition. This means that a WS should be trusted the first time during discovery (development) and then be consumed without the TN protocol being involved on the ensuing requests. Moreover, no standard protocols or languages have been defined, so the different proof of concept systems are unable to talk each other. Finally, adopting a TN approach would require that both parties reason and act according to a credential-based notion of trust. Other trust meanings are not supported. A first preliminary work on this direction has been proposed by Dragoni et al. in [41][47].

**Scenario 11**: Alice’s agent finds an interesting flight booking WS. It starts the TN protocol, disclosing step by step the required company credentials and finally it trusts it. Now the WS is inserted in the VTA. Whenever a VTA client user the service, the flight booking WS requires all the TN protocol to start over, that is not inefficient and useless because the service has already been trusted by Alice’s agent during the discovery.

### 3 Conclusions

As we verified in the previous section, there have been suggested many approaches for the automatic trust provision. Some of them are not directly addressed to a SOA environment, but they can still be adapted to it. The literature about that is growing, but, as it is possible to verify out of this analysis, WS trustworthiness provision is still an open challenge.

While investigating on studies in literature, it has been possible to derive shortcomings and advantages of the single approaches. They are summarized in the table 1. In order to optimize space and improve clarity, the plus and minuses are synthesized in few main classes, each one with its notation:

**Shortcomings**

- **NSR (i.e. New WS Ramp-up)**: refer to definition 6;
- **CD (i.e. Community Dependent)**: a community dependent system is affected by the community bootstrap issue (refer to definition 5);
- **NUR (i.e. New User Ramp-up)**: refer to definition 8;
- **HS (i.e. Hard Setup)**: this problem is connected to those approaches that require a big effort to be integrated in the real world;
- **UT (i.e. Unconditional Trust/Distrust)**: this issue is related to the approaches were the
user has to consume a service without any previous experience or evidence that the service is trustworthy. Or, the other way around, the user distrust the service unconditionally for the same reason;

**CE** (i.e. Centralized): refer to section 1.1;

**Advantages**

**PUTS** (i.e. Pre Use Trust Score): chances to obtain a trust score before using the service;

**UFS** (i.e. User Fitting Score): the WS trust score is also somehow related to the user personal “tastes” and habits.

There are also other specific pros and cons related to certain approaches that will be described directly in the table.

<table>
<thead>
<tr>
<th>Approach</th>
<th>Pluses</th>
<th>Minuses</th>
</tr>
</thead>
<tbody>
<tr>
<td>Direct experience</td>
<td>UFS → the most fitting score</td>
<td>CB, CD, CE → for those methodologies based on a centralized architecture</td>
</tr>
<tr>
<td>Social</td>
<td>Reputation</td>
<td>PUTS</td>
</tr>
<tr>
<td>Referrals</td>
<td>FUTS, rates coming from trusted peers</td>
<td>CB, CD</td>
</tr>
<tr>
<td>Matchmaker</td>
<td>FUTS, UFS, some community based methodologies provide for liars recognition</td>
<td>CB, CE → for those methodologies based on a centralized architecture, trust towards service moved to trust towards agent</td>
</tr>
<tr>
<td>Socio-Cognitive</td>
<td>Accurate trust computation, UFS</td>
<td>depending on the belief source: CE/NSR/CD/CE/NSR, CE → cognitive agents have to be conforming to a model</td>
</tr>
<tr>
<td>Trust &amp; Reputation</td>
<td>Trust &amp; Reputation</td>
<td>CE, NSR, CD</td>
</tr>
<tr>
<td>DE &amp; Reputation</td>
<td>Issues of the 2 constituent models mitigated</td>
<td>NSR, CD</td>
</tr>
<tr>
<td>Automated Trust Negotiation</td>
<td>UFS, PUTS, trust can ALWAYS be computed</td>
<td>CE, no standards defined, no WS aware (at the current state)</td>
</tr>
</tbody>
</table>

Table 1. Pluses and Minuses summary

In order to carry out this evaluation, few questions have been used as guidelines to judge each system:

**How does the trust score fit the user needs?** It would be better to build the trust score around the user profile, as the recommendation systems (2.2) do;

**Does the provider/consumer have to disclose any sensitive informations?** Some centralized approaches ask the user (be it either the provider, the consumer or both) to submit some personal information in order to improve the trust score computation. This is clearly something that should be avoided; users usually don’t want to unconditionally disclose sensitive details to a central authority.

**Can the user know how the trust is calculated?** Depending on the system architecture, the trust score might be calculated by a third party in a black box. The user may rather prefer to know how the service he is going to trust has been suggested.

**How does the community influence the trust score?** This issue mainly affects the social approaches (2.2): is the trust score depending on the size/quality of a community? In such a case, the main shortcoming would be the community bootstrap, i.e. how to create an initial community to kick-off the system.

**What is the trustworthiness of a brand new WS?** A new WS (i.e. it has been recently
deployed) needs a way to be “tried” even with no previous knowledge about it. The TN approaches seems to be a good starting point to address this issue.

**Does the user has to unconditionally trust/distrust certain services?** Whenever a user finds a WS, if the functional contract meets the user needs, there should always be a way to provide a trust score for that service, without leaving the user in the position of unconditionally trusting/distrusting the WS.

**How hard is the trust provisioning infrastructure to setup and maintain?** When designing a trust provisioning system it has to be taken into account both the effort needed to apply the system to the already existing SOC infrastructure, and the issues intrinsically related to the nature of a SOA-based system. A trust provisioning system is in charge of a great responsibility, and its robustness and scalability is a critical point in a SOA environment. Thus, for instance, it would be rather irresponsible to adopt a pure centralized architecture (that would be the single point of failure).

The analyzed approaches can be actually further on generalized to two big classes. The first one comprises the vast majority, i.e. the ones based on the either on direct experience of the consumer with the service, indirect experience (opinions on the service coming from someone trusted by the consumer) or a combination of them (hybrid). All of them has two key limitations in common: the user has to obtain trust from his own direct experience OR from the direct experience of someone else he trusts. It is usually safer to trust, for instance, 15 people saying that something is good instead of directly trusting something hoping it will be good. But still, this requires 15 people taking the risk to try. As correctly pointed out by Dragoni in [2]: *if someone does not take the risk of invoking an unknown service for the first time, then no one will be able to decide about the trustworthiness of the service before its invocation*. This class of approaches is based on a “soft trust” mechanism ( similar to the idea of “soft security” coined by Rasmusson et al. [48]) and they share the critical issue of service and community bootstrap. The rationale of the “soft trust” is that participants in a market collaborate each other in sharing information on other participants or services. Soft trust expect and even accept that there might be malicious services or service providers in the system. The idea is to identify them and prevent them from harming the other participants by means of collaboration and social interactions.

The other class of approaches, such like TN and Matchmakers, relies on a “hard” notion of trust: trustworthiness of a WS could be derived just from the a non-functional contract. They take into account the semantic of a WS, i.e. their security behaviour (e.g. access control rules, QoS features and so on). The recent Security-By-Contract (SxC) approach [41] might represent a good starting point for this purpose, because it takes into account the security behavior of a service instead of depending on the social control philosophy in the existing trust based approaches. Nevertheless, even the “hard trust” provisioning approaches studied in literature has a critical drawback: the lack of “fault recognition” capability. Everyone can provide a fake/wrong contract, be it due either to a malicious behaviour or to human distraction (or other unforeseeable problems). In this case the community help would turn useful. Moreover, it seems that these kind of approaches are still studied in a too theoretical fashion, without considering the practical issues related to WSs. For instance, the TN approaches seems to work fine assuming that the WS is consumed by a user directly invoking it. But in the real world, as
said in section 2.4, WSs are a “developer thing”. This means that they have to be trusted during the development time and then transparently consumed by clients unaware of the distributed nature of the system they are using.

Hybrid systems turned to be generally improving the constituent methods. Thus, a good direction to follow is probably to design a system capable of providing the features of both the macro-families: a “soft trust” system working along with a “hard trust” one should lead to a framework where the SOA developer can always evaluate automatically the available WSs for a given need and where a community is able to push unworthy WSs (and providers) aside. Finally, one of the major points of confusion concerning the current discussions about trust/trustworthiness in SOA is related to the meaning of terms. The studies analyzed in section 2, often use the word trust and the word trustworthiness with the same acceptance or with different meanings in different documents. The two terms have a precise meaning and trust should not be confused with trustworthiness. This should be probably the first issue to address in order define an acceptable solution.

References
42. Yu, T., Winslett, M., Seamons, K.E.: Supporting structured credentials and sensitive policies through interoperable strategies for automated trust negotiation. ACM Trans. Inf. Syst. Secur. 6 (February 2003) 1–42