Reputation management for distributed service-oriented architectures

C. Crapanzano, F. Milazzo, A. De Paola, G. Lo Re

In Proceedings of the Fourth IEEE International Conference on Self-Adaptive and Self-Organizing Systems Workshop (SASOW), 2010, pp. 160-165

Reputation Management for Distributed Service-Oriented Architectures

Calogero Crapanzano, Fabrizio Milazzo, Alessandra De Paola, and Giuseppe Lo Re
DINFO - Dipartimento di Ingegneria Informatica
Università degli Studi di Palermo, Italy
e-mail: {depaola, lore}@unipa.it

***Abstract*—Nowadays, several network applications require that consumer nodes acquire distributed services from unknown service providers on the Internet. The main goal of consumer nodes is the selection of the best services among the huge multitude provided by the network. As basic criteria for this choice, service cost and Quality-of-Service (QoS) can be considered, provided that the underlying Service-Oriented Architecture (SOA) be augmented in order to support the declaration of this information. The correct behavior of such new SOA platforms, however, will depend on the presence of some mechanisms that allow consumer nodes to evaluate trustworthiness of service providers. This work proposes a new methodology for discouraging antisocial behaviors of malicious service providers that declare QoS higher than the real one. The architecture is fully distributed over the network and emulates a decentralized hierarchical trusting authority capable of managing reputation values and of providing correct QoS assessments.**

***Index Terms*—Reputation Management; Distributed SOA; QoS-based Service Selection.**

I. Introduction

During the last few years, Internet applications have been greatly influenced by the introduction of innovative software architectures and new communication protocols for the construction of service-oriented network infrastructures. A Service-Oriented Architecture (SOA) is a software platform that describes the structure of service-oriented networks. Over the Internet, SOAs are typically implemented through the use of web services standards, and rely on a centralized approach, that requires the presence of a master node maintaining relevant information about network services and the relative providers. This approach suffers from well-known limits of centralized systems, i.e. lack of scalability and presence of a single point of failure. Distributed SOA (D-SOA) [1] represent an important evolution of classic SOAs and can overcome their limits using a hierarchical network structure, for distributing workload among network nodes. This architectural paradigm is well-suited in those scenarios in which trusting authority is implicitly distributed, for instance, as in Virtual Organizations, where agents belonging to different organizations interact by trading services. Each organization is responsible for its own resources and it is not feasible to entrust the management of all network resources to a centralized trusting authority.

In order to overcome this problem, we propose a hierarchical structure in which each organization represents a trusting authority for services held by its providers. However, the lack of a centralized trusting authority may encourage antisocial behaviors, such as declaring false QoS values.

This behavior is fully explained by game theory, according to which, the analysis of agent interactions in a real complex scenario cannot take into account the quality of being honest. On the contrary, each agent selects its own actions in order to achieve its maximum advantage, to the best of its knowledge, even if they cause damages to other agents [2]. In our scenario such an opportunistic behavior consists in the untruthful declaration of QoS values higher than the real ones, in order to guiltily promote inferior services. This consideration imposes that the traditional centralized trusting authority is replaced by a distributed one that offers equivalent functionalities.

The main contribution of our work is the definition of a distributed reputation management system that allows to set up a distributed trusting authority. The reputation management system evaluates the reliability of the service provider declarations and supports service consumers in the selection of truthful providers, filtering declared QoS values in order to obtain the actual ones by exploiting users' feedback.

According to the taxonomy presented in [3], our system can be defined as *personalized* and *decentralized*. It is decentralized because of the lack of a central entity managing information; rather, information on reputation are spread over the network. It is personalized because different nodes can have different reputation values for the same service provider; similarly, different nodes can receive different QoS assessments for the same service. Provider reputation is managed by exploiting consumers' feedback, released after service usage. The smaller the difference between declared and actual QoS, the greater the client satisfaction, with a consequent increase of the provider reputation. After an initial transitory phase, the system will converge toward an accurate estimate of the actual QoS values. Reputation values will be used in a mechanism of penalties and incentives, in order to allow consumer agents to identify malicious untruthful nodes.

The rest of the paper is structured as follows. In Sec. II other works presented in literature are described, as they contain some key concepts exploited in this work; Sec. III describes the proposed architecture, while details about the adopted policies for the management of QoS and reputation are provided in Sec. IV; Sec. V describes the gossip protocol that performs the reputation diffusion. Finally, Sec. VI reports the experimental results, and Sec. VII states some conclusions.

II. Related Work

The approaches presented in the scientific literature related to the problem of separating malicious and truthful nodes in SOAs involve different techniques, such as exploitation of users' feedback in order to produce single QoS estimate, single provider reputation estimate, or finally a hybrid QoS and reputation estimate.

In [4], the authors propose a model for reputation management in peer-to-peer networks. Information regarding the peers reputation is managed using ad hoc developed bayesian networks that periodically are exchanged among all peers. The reputation is updated by means of a reinforcement learning technique. This work, to the best of our knowledge, is one of the firsts which propose the adoption of reinforcement learning in order to model reputation. The decentralized system for reputation management and service selection proposed in [5] exploits monitor nodes for collecting QoS values of the service providers. QoS values are compared with users' feedback in order to filter out deceitful providers through clustering methods. This work presents a distributed form of trusting authority, however it is extremely inefficient because of the computational complexity of the adopted filtering algorithm. Authors of [6] propose the use of a Certification Authority (CA) in order to check that declared QoSs match their real values. Such approach does not take advantage from users' feedback and presents a centralized bottleneck that prevents the full scalability of the system. Also autors of [7] rely on certificates to guarantee agents' trustworthiness, but in this work none centralized certificate authority is proposed; on the contrary, authors propose a fully distributed solution that does not require to reveal agents' identity. A system capable of integrating users' feedback and reputation is proposed in [8]. Reputation is computed by a weighted sum of users' feedback, as a function of the feedback age. The approach is fully centralized, since user's feedback are stored in a centralized database. It is assumed that truthful service providers update QoS information and this last assumption is fully unrealistic for a real scenario. Authors of [9] proposes a trust network for a multi-agent system that exploits feedbacks individually provided by agents. Each agent provides its own belief of trustworthiness of the others, as a function of their past observed behaviors. The underlying Dempster Shafter theory of evidence allows to merge information coming from various agents and to cope with the lack of information. In [10] authors introduce the concept of service broker. The task of a broker is to seek those services in the network that better match user requirements, in order to maximize the customer utility function. The utility function for a service is computed under different conditions of load. Although a broker based approach may be useful in some architectures, the service broker computational load may be excessive in networks with a high density of providers. Finally, authors of [11] propose a system that expands traditional SOA with three additional components: QoS registries, Universal QoS matching and Web Service Broker. Brokers monitor the invoked services and compute a QoS value. Universal QoS matching compute the service QoS by a weighted sum of declared QoS, feedback QoS and monitored QoS, in which weights are directly proportional to the age of the information. Such approach, however, does not adopt any reputation management mechanism capable of inducing service providers to declare real QoS values.

Works discussed here presented some key concepts, like trusting authority, QoS estimate, reputation management and network monitors; nevertheless none of them merges all these aspects in a comprehensive approach. One of the contributions of our work is thus the integration of QoS estimate and reputation management in a single distributed mechanism. The proposed system has the advantage of effectively detecting malicious behaviors, maintaining the computational load low thanks to the exploitation of a hierarchy of authoritative nodes.

III. The Proposed Architecture

Our work proposes a system capable both of managing the providers reputation and of estimating real QoS values in a distributed SOA. The proposed architecture is well-suited for those scenarios in which trusting authority is implicitly distributed, since it allows the achievement of high degrees of guarantee for the QoS, still maintaining a full autonomy for the local resource management. Several real scenarios fall within this description, for instance Virtual Organizations [12] and Cloud Computing [13], whose main purpose is to dynamically coordinate different institutions in order to exchange services and advertise new ones. Adopting our QoS-based architecture, each institution can select the best services it needs on the basis of the QoS values estimates and discover malicious provider exploiting the provided reputation information.

A. System Architecture - Overview

From a logical point of view, the proposed D-SOA can be seen as a two-levels hierarchical network. Top-level subsystem is constituted by a set of *Super Nodes* forming an overlay network and acting as a distributed trusting authority. The low-level implements the service exchange subsystem and its components, called *Nodes*, are service consumers and providers. The main task of the distributed trusting authority is the monitoring of service exchange activity occurring at the low-level and the provision of updated information on both QoS and provider reputation for supporting service selection.

From a physical point of view, the whole network is partitioned into small clusters, called *domains*, as shown in Fig. 1. Each cluster is supervised by a Super Node which is responsible for monitoring the activities of all Nodes belonging to its domain. More precisely, Super Nodes are actively involved in the initial service negotiation phase and in the final phase of users' feedback management, whilst the actual service exchanges occur through direct connections among Nodes.

B. System Architecture - Functional View

Super Nodes overlay network implements the distributed trusting authority by evaluating the reputation of service providers. Each Super Node maintains a reputation value for

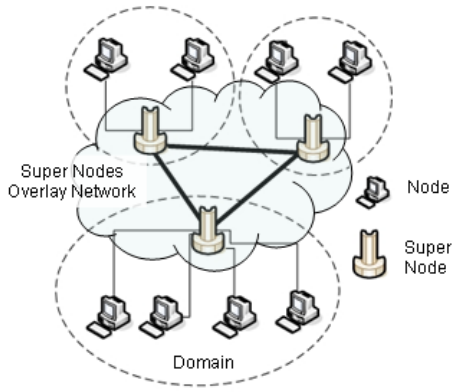

Fig. 1. Clustered hierarchy organization of *Super Nodes* and *Nodes*.

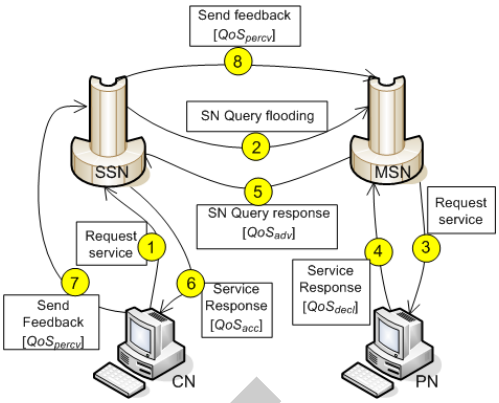

Fig. 2. Event flow generated by a service request.

each Node in its domain and also, in order to maintain links to different domains, a reputation value for any neighbor node in the overlay network.

The neighborhood reputation allows Super Nodes to filter QoS information as a function of the reputation of the node which provides it. This filtering phase allows service consumers to correctly select the best services matching with their requirements of quality and cost.

In order to provide a more detailed description of our architecture, the event flow generated by a service request is analyzed and shown in Fig. 2.

We distinguish four roles:

- *Consumer Node*: the service consumer (`CN` in Fig. 2);
- *Provider Node*: the service provider (`PN`);
- *Seeker Super Node*: the super node that is the cluster-head of the cluster hosting the Consumer Node (`SSN`);
- *Manager Super Node*: the super node that is the cluster-head of the cluster hosting the Provider Node (`MSN`).

When a CN looks for a given service, it sends a query to its SSN (1) that, in turn, forwards it to its Super Node neighbors in the overlay network (2). We assume, without loss of generality, that Super Nodes form a fully connected overlay network. Under this assumption, all Super Nodes in the network can reply to the query. For the sake of simplicity, the assumption also allows us to disregard problems related to the query routing that do not fall within the issues addressed by this work. When a Super Node receives a query, it performs a local search for the services provided by Nodes in its domain (3). Each Node replies to the local query declaring the updated QoS values for the requested service, QoS_{decl} (4). In this phase, these Nodes act as PNs. The Super Node replies to the the SSN with a list of services matching the query, enriched also by QoS information (5). In this phase, the queried Super Node acts as trusting authority for QoS information, thus playing the role of MSN. In its guarantor role, each MSN has also the capability of modifying QoS values. Hence, the SSN receives from a certain number of MSN lists of services coupled with respective QoS values advertised by respective MSNs, QoS_{adv}. A comprehensive merged list is then forwarded by SSN to the CN (6). Here, also the SSN has the capability of fixing the received QoS values thus producing the final accepted QoS values, QoS_{acc}. The CN selects a service out from the received list, and as final step closing the loop it determines a feedback value by estimating the perceived QoS, QoS_{percv}, to be send back to the SSN (7). As a function of the received feedback, the SSN is thus able to update its reputation estimate of the MSN. This update is performed according to a function which takes into account the similarity between advertised QoS and perceived QoS. Finally, the SSN forwards the same feedback value to the MSN (8), in order to enable it to apply the same procedure to update the reputation of the PN in its domain.

IV. QoS and Reputation Management Policy

The system behavior heavily depends both on the policies adopted for the QoS and reputation management and on the service selection methods. For this reason, in order to provide a full specification of the system, we shall provide the description of the following functionalities:

- Service selection performed by CN after receiving the service list (step 6);
- MSN reputation management as performed by SSN after receiving users' feedback (step 7);
- PN reputation management as performed by the MSN after receiving users' feedback (step 8).

All the above policies exploit the reinforcement learning mechanism as their driving principle.

A. The Adopted Reinforcement Learning Model

Reinforcement Learning (RL) [14] is a branch of Machine Learning, modeling how agents learn which actions to perform with the aim of maximize a score function, based on the results of past interactions with the environment. The RL model assumes that, after each interaction with the environment, the playing agent obtains a reward for the performed action. Such rewards constitute the input data of a trial-and-error learning mechanism whose goal is the generation of the best situation-action mapping to be considered for maximizing the average reward. In order to select the action to be performed, an optimal trade-off between the *exploitation* of the acquired knowledge and the *exploration* of not-yet-evaluated solutions must be achieved. The former criterion involves that the agent

would choose the best action given its current state, whereas the latter implies that the agent would also choose sub-optimal actions in order to explore new outcomes.

We identify the learning agents with the network nodes, and for each agent its environment is represented by all the other nodes. In order to perform the reputation management, QoS estimate, and service selection, we adopted the *Q-learning* [15], among all the proposed RL techniques, because it considerably simplifies the formalization of the learning algorithm and it comes with a formal proof of its early convergence. In such method, the average utility of performing an action a in a state s, referred as $Q(s_t, a_t)$, is updated as a function of the past estimate and of the reward r_{t+1} obtained after the agent-environment interaction, according to the following equation:

$$\begin{aligned} Q(s_t, a_t) &\leftarrow (1-\alpha)Q(s_t, a_t)+ \\ &+ \alpha[r_{t+1} + \gamma \max_a Q(s_{t+1}, a)], \end{aligned} \quad (1)$$

where $Q(s_t, a_t)$ is the current estimate of the utility obtained by performing the action a_t in the state s_t, s_{t+1} is the new state in which the agent transits after the action performance, r_{t+1} is the obtained reward, and $\max_a Q(s_{t+1}, a)$ is the maximum reward obtainable in the new state. The α and γ parameters, both ranging in $[0, 1]$, control the learning mechanism, and represent respectively the learning rate and the discount factor. The former determines the weight of new information with respect to the past history, and the latter determines the influence of future rewards. Based on the $Q(s_t, a_t)$ values, the agent selects the action to be performed with a technique known as *reinforcement comparison*, according to which, each action can be selected with a probability π directly related to its estimated average reward, computed as follows:

$$\pi_t(a, s) = Pr\{a_t = a | s_t = s\} = \frac{e^{Q(s_t,a_t)/\tau}}{\sum_a e^{Q(s_t,a)/\tau}}. \quad (2)$$

Such a selection mostly stresses the choice of the best action thus enabling the exploitation; however, since the probability to select sub-optimal actions is never 0 , it also allows the exploration. High values for the τ parameter, *temperature* in the Boltzmann distribution, make the actions quite equiprobable, while low values make that a small difference in action utility correspond to a big difference in action selection probability.

B. QoS Filtering and Service Selection

In response to the query for a service, the Consumer Node receives a list of services matching the query parameters. These services are associated to some QoS information, advertised by MSNs and filtered by the SSN. QoS information filtering is performed by the SSN on the basis of its reputation value associated to the MSN that provided such information. If rep represents the reputation value of the MSN providing the QoS declaration, rep_{max} the maximum reputation value of all neighbor Super Nodes, and QoS_{adv} the QoS value advertised by the MSN, the filtering rule which determines the accepted QoS, QoS_{acc}, can be written as:

$$QoS_{acc} = QoS_{adv} * \frac{rep}{rep_{max}}. \quad (3)$$

After its filtering activity, the SSN forwards the modified service list to the CN, in order to support it in the selection of best services. Selection is performed through reinforcement comparison method described in eq. 2, where the selection of a service corresponds to an action, and the action reward corresponds to the QoS of the selected service.

Through this selection mechanism, the CN acts with the direct goal of maximizing its own utility and with the indirect effect of penalizing malicious PNs. Namely, a low reputation value will correspond to low accepted QoS values and definitely this will led to less sold services.

C. Manager Super Nodes Reputation Management

After a CN uses a service, it replies its SSN, with a feedback value expressing the perceived QoS. The SSN exploits this information in order to update the MSN reputation. This update operation takes into account the gap between QoS_{acc} and QoS_{percv}. In such a phase the SSN may choose among three possible actions: it may increase, decrease, or confirm its MSN reputation. Intuitively, if the accepted QoS is similar enough to the QoS perceived by the CN, the current estimate of the MSN reputation value can be considered correct and then confirmed. Vice versa, if the filtered QoS value does not correspond to the perceived one, it is more appropriate to update the reputation estimate.

In order to select the best action to be performed, an ad-hoc reputation-learning subsystem was designed. All the possible states of the subsystem represent the set of possible reputation values for the MSNs in the neighborhood of the SSN; the subsystem goal is to learn the utility value of each action in all possible states. In this context, the utility value is a function of the similarity between filtered and perceived QoS values.

In the current state, for each possible action (in short: $incr$, $decr$, $conf$), the SSN evaluates which QoS value would have transferred to the CN, using eq. 3; for each of these three hypothetical values, the SSN evaluates the difference between perceived QoS and hypothetical filtered QoS, QoS_{acc_hyp}. Finally, this hypothetical error, err_{hyp}, is compared to the actual one, err_{act}, in order to obtain the reward r for all the possible actions, according to the following equations:

$$\begin{aligned} err_{act} &= |QoS_{percv} - QoS_{acc}|, \\ err_{hyp} &= |QoS_{percv} - QoS_{acc\_hyp}|, \\ r &= err_{act} - err_{hyp}. \end{aligned} \quad (4)$$

Obviously, the action of confirming the reputation of the MSN has a null reward.

The average utility for all actions is updated using the Q-Learning method as described in Sec. IV-A, as a function of the computed rewards. As regards the current state, represented by the current reputation of the MSN, the rule for updating actions' utility is the following:

$$\begin{aligned} Q(rep, incr) &\leftarrow (1-\alpha)Q(rep, incr)+ \\ &+ \alpha[r_{incr} + \gamma \max_a Q(rep+1, a))], \\ Q(rep, decr) &\leftarrow (1-\alpha)Q(rep, decr)+ \\ &+ \alpha[r_{decr} + \gamma \max_a Q(rep-1, a)], \\ Q(rep, conf) &\leftarrow (1-\alpha)Q(rep, conf) \\ &+ \alpha[r_{conf} + \gamma \max_a Q(rep_t, a)]. \end{aligned} \quad (5)$$

In summary, when a SSN receives a feedback from a CN, it performs the following actions:

1) Compute the reward values, r_{incr}, r_{decr}, r_{conf} (eq. 4);
2) Evaluate the effects of possible actions for updating MSN reputation, by computing their utility values, $Q(rep, incr)$, $Q(rep, decr)$, $Q(rep, conf)$, (eq. 5);
3) Select the action to be performed through the reinforcement comparison method (eq. 2);
4) Update the reputation of the MSN according to the selected action.

D. Provider Nodes Reputation Management

The fact that SSN evaluates the reputation of MSN by estimating the reliability of the QoS information they advertise, represents, ultimately, the reason why a MSN that does not correctly certify the reputation of the PN in its own domain may experience a reduction of reputation, since it is not able to discover malicious behaviors. In order to avoid other Super Nodes discredit, each Super Node maintains the reputation values of all the PN belonging to its domain, with the aim to penalize them whenever they declare incorrect QoS values. The information necessary to manage the reputation of the PN is obtained from users' feedback that are forwarded by the SSN. Such a management policy mirrors the management policy of MSN reputation carried out by the SSN, and described in Sec. IV-C. Reputation values of the PN are here used by MSN to filter at the origin advertised QoS values, with a mechanism equivalent to that described in Sec. IV-B.

V. Gossip Protocol for Reputation Diffusion

Achieving a homogeneous evaluation criterion in the distributed trusting authority represents an important goal of our work. A problem present in the Super Node overlay network, is that local reputation estimates of Super Nodes may significantly differ among the overlay Nodes. This is mainly due to the fact that the local reputation estimate of a Super Node is performed only after the usage of services provided by its domain. This means that if Super Node SN_i does not require services guaranteed by Super Node SN_j, it will not update its local value of reputation at all. Therefore, $r_i^j(t)$, i.e. the reputation of Super Node SN_j as estimated by Super Node SN_i, may differ substantially from the one estimated by Super Node SN_k.

In order to achieve a substantial agreement in the Super Node overlay network, we propose a gossip-based protocol, for reputation diffusion, based on a previous work [2]. Such a protocol aims to diffuse information reliability, so as to obtain a view of the network as more homogeneous as possible. According to this protocol, each Super Node periodically sends to its neighbors its reputation table. A Super Node that has received the reputation tables from its neighbors, merges received information into its own reputation estimate with a weight proportional to the reputation of the source. The reputation merging is thus performed only if the source is considered reliable, that means, only if its reputation value exceeds a given threshold. Given the Super Node SN_i, the set K_i of its reliable neighbors is a subset of the adjacent Super Nodes, and is calculated according to the following equation:

$$K_i = \{k : r_i^k(t) \geq \rho\}. \tag{6}$$

Exploiting the reputation tables received from its reliable neighbors K_i, SN_i updates the reputation value regarding Super Node SN_j, according to the following equation:

$$r_i^j(t+1) = (1-\beta) * r_i^j(t) + \beta * \frac{\sum_{k \in K_i} r_k^j(t) * r_i^k(t)}{\sum_{k \in K_i} r_i^k(t)}. \tag{7}$$

The β coefficient tunes the weight of the gossip information with respect to the local estimate; the ρ threshold in eq. 6 expresses the trustiness degree of a Super Node in its neighbors.

VI. Experimental Results

In order to evaluate the system behavior, we performed a wide set of simulations, through an ad-hoc developed simulator. We report the results of experiments devoted to highlight how the reputation management system motivates MSN to correctly advertise the reputation of their domain PNs, and how PNs are compelled to declare true QoS values.

A. Advantages from a correct reputation management

The first experiment focuses on the usefulness deriving to MSNs from a correct management of the PN reputation. In our setting the simulated network is composed of 150 nodes spread over 11 domains, containing the same number of honest PNs, malicious PNs and CNs. Each SN correctly manages the reputation only for a part of the malicious PNs belonging to its domain, and masquerades for the other part. The experiment consists of 10 simulations of 1000 steps and the results were averaged over all simulations. Fig. 3 shows that MSNs that correctly manage the reputation of greater percentages of PNs achieve a clear advantage, since they obtain high values of reputation, whereas the reputation of malicious MSNs decreases quickly over the time. Namely, since SSNs must provide the most accurate QoS estimates to CNs, they dramatically reduce the MSN reputation until the accepted QoS values, filtered according to eq. 3, match the perceived ones.

B. Detection of Malicious Provider Nodes

In order to effectively detect malicious PNs, a MSN should assign different reputation values to PNs as a function of declared QoS values. The second experiment aims to prove that the reputation management policy provides MSNs with this capability. The setting differs from the previous experiment since all MSNs correctly manage the reputation of PNs. Fig. 4 shows the average value of reputation PNs, aggregated by the probability of declaring false values. The reputation scores of malicious PNs decrease over the time, for the same reason adduced in the previous experiment. This leads to an important conclusion: malicious behaviors are always detected, either by the decrease of the PN reputation in the opinion of its MSN, or

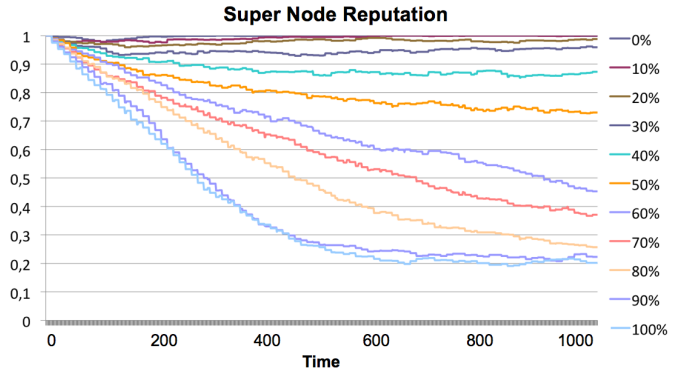

Fig. 3. Comparison between the reputation values of MSNs that masquerade for different percentages of malicious PNs.

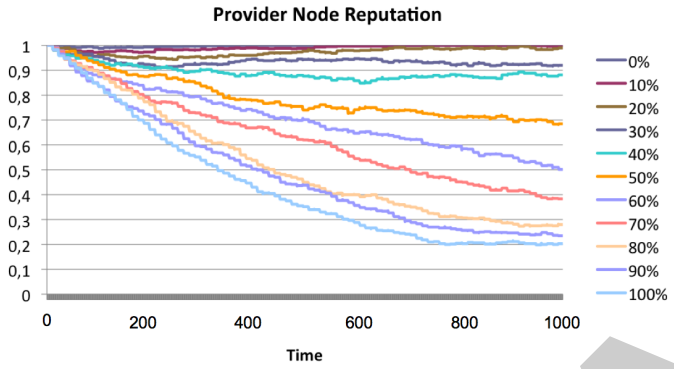

Fig. 4. Comparison between the reputation values of PNs that provides false QoS values with different probabilities.

by the decrease of MSN reputation when it does not correctly advertise its PN reputations.

C. The Economic Drawback for Malicious Provider Nodes

A reputation management system able to detect malicious nodes can actually lead PNs to declare true QoS values only if this detection corresponds to some economic drawback for malicious providers. Such a deterrent can originate only from a reduction of sold services. In this experiment, we use the same basic setting of of the previous ones. Results shown in Fig. 5 prove that, after a transitory phase, during which services provided both by malicious and truthful PNs are chosen with the same percentage, a drastic reduction of the percentage of selected services (that do not reach 5% for PN declaring false values more often than 40%) is determined by the decrease of reputation values for malicious PNs.

VII. Conclusions

D-SOAs represent a novel architectural paradigm well-suited in scenarios in which limits of classical SOAs, caused by their intrinsically centralized nature, constitute severe disadvantages. D-SOA can exploit additional parameters, such as QoS, in order to support consumers in the selection of the best services. Unfortunately the lack of a centralized supervising entity favors antisocial behaviors. A hierarchical reputation management system was proposed in order to effectively detect and penalize malicious behaviors. Our system is based on

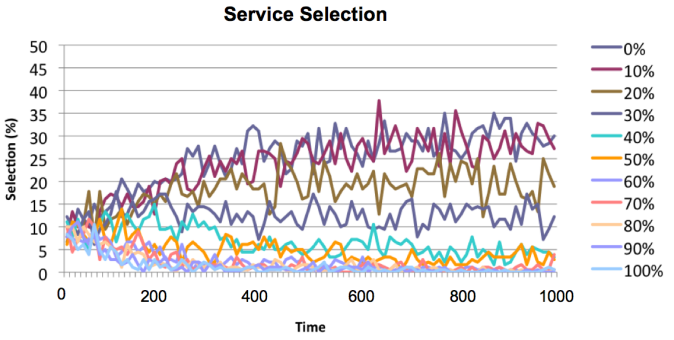

Fig. 5. Comparison between the average percentage of selected services provided by PNs with different probability of being dishonest.

decentralized Reinforcement Learning approach, and allows Consumer Nodes to learn the best service in order to maximize the perceived QoS and motivate Provider Nodes to honestly behave, declaring true QoS.

References

[1] F. Banaei-Kashani, C. Chen, and C. Shahabi, "WSPDS: Web Services Peer-to-peer Discovery Service," in *Proceedings of the International Conference on Internet Computing*, 2004, pp. 733–743.

[2] A. De Paola and A. Tamburo, "Reputation Management in Distributed Systems," in *3rd International Symposium on Communications, Control and Signal Processing (ISCCSP)*, 2008, pp. 666–670.

[3] J. Vassileva and Y. Wang, "A Review on Trust and Reputation for Web Service Selection," in *27th International Conference on Distributed Computing Systems Workshops (ICDCSW'07)*, 2007.

[4] Y. Wang and J. Vassileva, "Trust and Reputation Model in Peer-to-Peer Networks," in *IEEE Conference on P2P Computing*, 2003.

[5] L.-H. Vu, M. Hauswirth, and K. Aberer, "QoS-based Service Selection and Ranking with Trust and Reputation Management," in *On the Move to Meaningful Internet Systems*, 2005, pp. 466–483.

[6] S. Ran, "A model for web services discovery with QoS," *ACM SIGecom Exchanges*, vol. 4(1), pp. 1–10, 2004.

[7] Y. Mass and O. Shehory, "Distributed trust in open multi-agent systems," *Trust in Cyber-societies*, pp. 159–174, 2001.

[8] Z. Xu, P. Martin, W. Powley, and F. Zulkernine, "Reputation-enhanced QoS-based web services discovery," in *IEEE International Conference on Web Services, 2007. ICWS 2007*, 2007, pp. 249–256.

[9] B. Yu and M. Singh, "An evidential model of distributed reputation management," in *Proceedings of the first international joint conference on Autonomous agents and multiagent systems (AAMAS)*. ACM, July 2002, pp. 294–301.

[10] D. A. Menasce and V. Dubey, "Utility-based QoS Brokering in Service Oriented Architectures," in *IEEE International Conference on Web Services*, 2007, pp. 422–430.

[11] G.-Q. Liu, Z.-L. Zhu, Y.-Q. Li, D.-C. Li, and J.-C. Cui, "A New Web Service Model Based On QoS," in *International Symposium on Intelligent Ubiquitous Computing and Education*, 2009.

[12] T. Norman, A. Preece, S. Chalmers, N. Jennings, M. Luck, V. Dang, T. Nguyen, V. Deora, J. Shao, and W. Gray, "Agent-based formation of virtual organisations," *Knowledge-Based Systems*, vol. 17, no. 2-4, pp. 103–111, 2004.

[13] R. Buyya, C. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems*, vol. 25, no. 6, pp. 599–616, 2009.

[14] R. Sutton and A. Barto, *Reinforcement Learning: An Introduction*. The MIT press, 1998.

[15] C. Watkins and P. Dayan, "Q-Learning," *Machine Learning*, vol. 8, no. 3, pp. 279–292, 1992.