In E-Business Application for Product Development and Competitive Growth: Emerging Technologies, pp. 333-343, IGI Global

Web Services Communities: From Intra-Community Coopetition to Inter-Community Competition

Zakaria Maamar
Zayed University, Dubai, UAE
Philippe Thiran
University of Namur, Namur, Belgium
Jamal Bentahar
Concordia University, Montreal, Canada

Abstract
This chapter discusses the structure and management of communities of Web services from two perspectives. The first perspective, called coopetition, shows the simultaneous cooperative and competitive behaviors that Web services exhibit when they reside in the same community. These Web services offer similar functionalities, and hence are competitive, but they can also cooperate as they share the same savoir-faire. The second perspective, called competition, shows the competition that occurs not between Web services but between their communities, which are associated with similar functionalities. To differentiate such communities, a competition model based on a set of metrics is discussed in this chapter.
Keywords. Web services communities, coopetition, competition, reputation.

1. Introduction

For the World Wide Web Consortium, a Web service ``*is a software application identified by a URI, whose interfaces and binding are capable of being defined, described, and discovered by XML artifacts and supports direct interactions with other software applications using XML-based messages via Internet-based applications*''. For the last few years, the development pace of Web services has been impressive (Di Martino, 2009; Maamar et al., 2008; Maaradji et al., 2010; Khosravifar et al., 2010a). On the one hand, several standards have been developed to deal for example with Web services definition, discovery, and security. On the other hand, several projects have been initiated to examine among other things Web services composition, personalization, and contextualization.

Nowadays, competition between businesses does not stop at goods, services, or software products, but includes as well Web services. Independent providers develop several Web services that sometimes offer the same functionality such as weather forecast and currency exchange. It is reported in (Bui, 2005) that although Web services are heterogeneous, the functionalities they offer are sufficiently well defined and homogeneous enough to allow for market competition to happen. To ease and improve the process of Web services discovery in an open environment like the Internet, we gather similar Web services into communities (Maamar et al., 2009). Acknowledging the efforts that service engineers need to put into designing and managing communities, we put forward guidelines that define how to specify and set up a community, how to manage the Web services that reside in a community, and how to reconcile conflicts within a community and between communities. In a community, the Web services are simultaneously in competition and in cooperation, i.e., they compete to participate in composite Web services since they all offer the same functionality with different non-functional properties (or QoS) and at the same time cooperate when they substitute for each other in case of failure.

As several communities of Web services come online and in line with today's economy featured by competition, there, also, exist relationships between communities that offer the same functionality. This increases the competition among communities of Web services and potential users of Web services. On the one hand, providers are interested in finding the communities that give better exposure to their respective Web services. On the other hand, users are interested in binding to the best communities that host the appropriate Web services as per their respective needs. The development of

a reputation model is deemed appropriate for both parties.

The remainder of this chapter is organized as follows. The next section introduces the definitions of some concepts upon which communities of Web services are built. The intra-community coopetition and inter-community competition are discussed after that. Finally some concluding remarks are drawn.

2. Some definitions

Community. It means different things in different settings. In Longman Dictionary, community is ``*a group of people living together and/or united by shared interests, religion, nationality, etc.*''. In the field of knowledge management, communities of practice constitute groups within (or sometimes across) organizations who share a common set of information needs or problems (Davies, 2003). Communities are not formal organizational units but networks with common interests and concerns. When it comes to Web services, Benatallah et al. define community as a collection of Web services with a common functionality although these Web services have distinct non-functional properties (Benatallah, 2003). Medjahed and Bouguettaya use community to provide an ontological organization of Web services sharing the same domain of interest (Medjahed, 2005). Medjahed and Atif use community to implement rule-based techniques for comparing context policies of Web services (Medjahed, 2007). Maamar et al. define community as a means to provide a description of a desired functionality without explicitly referring to any concrete Web service that will implement this functionality at run-time (Maamar et al., 2009). Finally, Wan et al. define communities of Web services as virtual spaces that can dynamically gather different Web services having complementary or related functionalities (Wan et al., 2010).

Reputation. It is ''*the opinion (more technically, a social evaluation) of the public toward a person, a group of people, or an organization*'' (Wikipedia). Reputation, besides other selection criteria, has been widely used for evaluating and ranking participants in social networks, online collaborations, agent-based systems, or in e-business platforms (like e-Bay). Nowadays, opinions and user ratings are no longer sufficient for assessing the reputation of computer systems as Elnaffar stresses out in (Elnaffar 2006). Opinions/Ratings are subjective and can be easily tampered. A reputation system that solely relies on the temporal perspective of humans (i.e., clients) can expose the system, intentionally or unintentionally, to dishonest ratings caused by the following types of users:

- Emotional reactors: clients who give non-subjective, inaccurate ratings influenced by some personal (could be temporal) issues with the system being used.
- Bad mouthers: clients who unfaithfully exaggerate by giving negative ratings to service providers. These clients might plan to collude with the competitors of these providers.
- Ballot stuffers: clients who unfaithfully exaggerate by giving positive ratings to service providers. Like for bad mouthers, these clients plan to collude with these providers.

Coopetition. It is kind of combination between cooperation and competition (Bengtsson and Kock, 2000). Simultaneously, actors are engaged in cooperative and competitive relationships. Actors have, on the one hand, conflicts of interests (competition) and, on the other hand, common interests (cooperation). In a shopping mall retailers that display the same merchandise exhibit a competitive behavior, but at the same time cooperate to attract the maximum number of customers since they are in the same mall. They aim at providing customers with more choices (also known as externalities) and compete since each retailer is a profit maximizer.

3. Intra-community coopetition

In a community, competition and cooperation coexist among Web services. Competition appears because each Web service aims at maximizing its participation rate in Web services composition. Cooperation takes place because Web services need each other to sustain the growth of their community by for example maintaining a good reputation level and substituting each other in case one of them fails. Web services are part of a two-side market (Roson, 2005). The benefits of one side from interacting through the community depend on the size of the other side that consists of Web services contributing together to the attractiveness of their community. As a result, a community must make itself attractive to users by encouraging and managing the competition and cooperation (i.e.,

coopetition) of its Web services. The community therefore cares about the interactions between the two sides and the management of its members.

3.1 Architecture

Fig. 1 represents the architecture of Web services communities. The components of this architecture are the internal structure of communities, providers of Web services and UDDI registries (or any type of registry like ebXML). Communities are established and dismantled according to specific scenarios and protocols that are detailed in Section 3.2. UDDI registries receive advertisements of Web services from providers for posting purposes. Fig. 1 offers some characteristics that need to be stressed out. First, the common way to describing, announcing, and invoking Web services is still the same as described in the standard service-oriented architecture although Web services are now associated with communities. Second, the regular facilities that UDDI registries offer are still the same; no extra facilities are required to accommodate communities' needs. Finally, the selection of Web services out of communities is transparent to users and independent of the way they are gathered into communities. Two communities of Web services are shown in Fig. 1. They could have airfare quotation and hotel booking as examples of functionalities, respectively. A master component always leads a community. This master could itself be implemented as a broker Web service (like shown in Fig. 1) for compatibility purposes with the rest of Web services in a community, which are now denoted as slaves. Master-Slave Web services relationship in a community is regulated using the well-known Contract Net protocol (Smith, 1980) (*CNProtocol*). However, more advanced protocols involving negotiations based on argumentation can also be used (Bentahar et al. 2007).

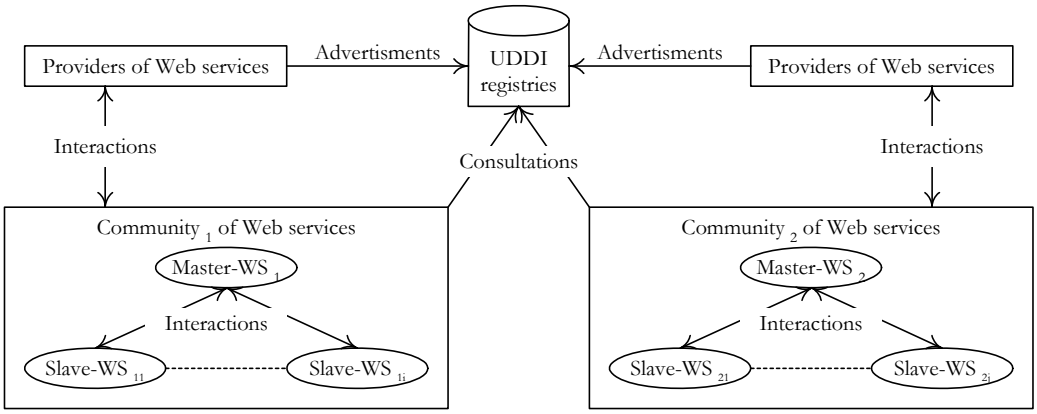

Figure 1 Architecture of single Web services communities per functionality

One of the responsibilities of the master Web service is to attract Web services to make them sign up for its community using rewards (Maamar et al., 2009). As a result, the master Web service checks regularly the UDDI registries so that it is kept updated about the latest changes like new advertisements in their respective contents. The master also checks the credibility of the potential members, so only trustful Web services are invited (Khosravifar et al., 2010b). More responsibilities of the master Web service include nominate the slave Web service out of several peers to participate in a composite Web service, and run the *CNProtocol* to nominate this Web service.

In a community, the master Web service is designated in two different ways. The first way is to have a dedicated Web service playing the role of master for the time being of a community. Since the master Web service never participates in any composition, it is only loaded with community management mechanisms like Web services attraction and retention. The second way is to identify a Web service from the list of available Web services to act as a master. This identification happens on a voluntary basis or by running an election process among the Web services. Because of the temporary no-participation restriction of a master Web service in compositions, the nominated Web service is

compensated (Bentahar et al., 2008). The call for elections in a community takes place regularly and in a democratic way, so that the burden on the same Web services to lead a community is either minimized or avoided.

3.2 Operation

Community operation is about developing a new community, dismantling an existing community, attracting new Web services to a community, retaining existing Web services in a community, and selecting slave Web services in a community to take part in a composition.

Community development. A community gathers similar Web services using a two-step process. The first step is to define the functionality, e.g., flight booking, of the community. The second step is to deploy the master Web service that will lead the community and take over the aforementioned responsibilities. One of them is to invite and convince Web services to sign up for its community. The survivability of a community, i.e., to avoid dismantlement, depends on the performance of the community, which in turn depends to a certain extent on the performance of the existing Web services in this community. Another responsibility is to check the credentials (e.g., announced QoS) of Web services before they are admitted in a community. This checking has a dual advantage: boost the security level among the peers in a community and enhance the trustworthiness level of a master Web service towards the slave Web services it manages. The first advantage avoids dealing with malicious Web services that could attempt to alter other peers' data and behaviors. The second advantage shows the reliance of the master Web service on the slave Web services in completing the prescribed operations. Enhancing the security of a community is an important factor that contributes towards its reputation (Khosravifar et al., 2010a). Such a reputation is fundamental to attract both new Web services to sign up and users to request Web services (Elnaffar, 2008). It should be noted that slave Web services could turn out "lazy"[i] after joining a community, which calls for their immediate ejection from the community.

Dismantling a community happens upon request from the master Web service. This one oversees the events in a community such as arrival of new Web services, departure of some Web services, and sanctions on Web services because of misbehavior. When a master Web service notices that the community has poor performance over a certain period of time, the community designer can decide to dismantle it. This performance can be measured using the number of Web services that are registered in the community and the number of participation requests in composite Web services. Ultimately, the performance over a period of time $[t_1, t_2]$ can be measured from an economical perspective as follows:

$$Performance^{[t_1,t_2]} = Incomes^{[t_1,t_2]} - Rewards^{[t_1,t_2]}$$

where incomes are generated from the participation in composite Web services and rewards are expenses used to establish and manage the community by attracting and retaining good Web services. Consequently, the community has a good performance when the number of active Web services is high and close to the number of its members. However, the performance is poor when the number of active Web services is low compared to the number of registered Web services.

Web services attraction and retention. Attracting new Web services to and retaining existing Web services in a community fall into the responsibilities of the master Web service. A community could vanish if the number of active Web services that reside in it drops below a certain threshold that is set by the community designer.

Web services attraction makes the master Web service consult regularly the different UDDI registries looking for new Web services. When a candidate Web service is identified based on the functionality it offers, the master Web service interacts with its provider (Fig. 1). The purpose is to ask the provider to register its Web service with the community of this master Web service. Some arguments to convince the provider include high participation-rate of the existing Web services in composite Web services (it shows the visibility of the community and the reputation of Web services (Maximilien,

2002)), short response-time when handling user requests, and efficiency of the security mechanisms against malicious Web services (Khosravifar et al., 2010a).

Retaining Web services to remain committed to a community for a long period of time is a good indicator of the following elements:

- As mentioned earlier, although Web services in a community are competitive, they expose a cooperative attitude (i.e., they are coopetative). For instance, Web services have not been subject to attacks from peers in the community (because all Web services would like to participate in composition scenarios, some of them could try to make other peers less competitive by illegally altering their execution properties). This backs the security argument that the master Web service uses again to attract Web services and convince their providers.
- A Web service provider is satisfied with its participation rate in composite Web services. This is in line with the participation-rate argument that the master Web service uses to attract new Web services.
- Web services are, through the master Web service, aware of some peers in the community that could replace them in case of failure, with less impact on the composite Web services in which they are involved.

Web services attraction and retention shed the light on a third scenario, which is Web services being invited to leave a community as briefly reported earlier. A master Web service could issue such a request upon assessment of the following criteria:

- The Web service is unreliable. On different occasions the Web service failed to participate in composite Web services due to recurrent operation problems.
- The credentials of the Web service were "beefed up" to enhance its participation opportunities in composite Web services. Large differences between a Web services' advertised QoS and delivered QoS indicate performance degradation (Ouzzani, 2004).

Web services selection. In a community, selection of Web services to participate in composition rely on the Contract-Net protocol, namely job contracting and subcontracting between two types of agents known as initiator (master Web service) and participant (slave Web service). At any time an agent can be initiator, participant, or both. The sequence of steps in the contract-net protocol, which we slightly extend, is as follows: (1) initiator sends participants a call for proposals with respect to a certain job to carry out; (2) each participant reviews the call for proposals and bids if interested (i.e., feasible job); (3) initiator chooses the best bid and awards a contract to that participant; and (4) initiator rejects other bids.

Mapping the contract-net protocol onto the operation of a community occurs as follows. When a user (through some assistance (Schiaffino, 2004)) selects a community based on its functionality, the master Web service of this community is contacted in order to identify a specific slave Web service that will implement this functionality at run-time. The master Web service sends all slave Web services a call for bids (CNStep 1). Prior to getting back to the master Web service, the slave Web services assess their status by checking their ongoing commitments in other compositions and their forthcoming maintenance periods (Maamar, 2006) (CNStep 2). Only the slave Web services that are interested in bidding inform the master Web service. This latter screens all the bids before choosing the best one (CNStep 3)[ii]. The winning slave Web service is notified so that it can get itself ready for execution when requested (CNStep 3). The rest of the slave Web services that expressed interest but were not selected, are notified as well (CNStep 4).

4. Inter-community competition

In this section, we extend our previous discussion by considering that a functionality can be offered by more than one community. Hence, both providers and users have to decide whether or not to use or subscribe to a specific community. A party considers the incentive provided by the other party. For providers, a major incentive is the participation rate that the community can offer to them. For the users, the main factor is their satisfaction about the provided service. We argue in this chapter that these incentives can be captured through reputation.

4.1 Architecture

Fig. 2 depicts an augmented Web services community architecture that takes into consideration the notion of community reputation. As shown, the architecture consists of the following main components:

- UDDI registry: In addition to Web services' descriptions that are posted on the UDDI descriptions of available communities are posted, as well.
- User Agent: It is a proxy between the user and other parties namely, UDDI registry, Web services communities, and the reputation system.
- Provider Agent: Akin to the user agent, a provider agent is a proxy between the provider and other parties, namely, UDDI registry, Web services communities, and the reputation system.
- Reputation System: It maintains a repository of run-time operational data, called logs below, that are needed to compute the performance metrics of a community, and ranks communities by their reputation.

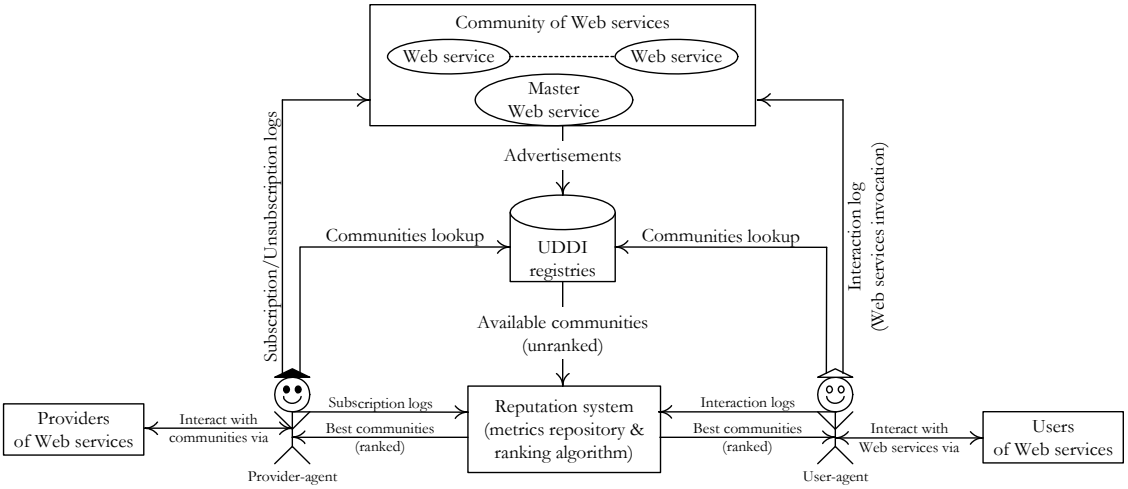

Figure 2 Architecture of multiple Web services communities per functionality

To compute reputation from the perspective of users or providers, user agents should intercept the following time-stamped logs for each event that happens during the interaction with the community

1. Received Requests log tracks all requests directed to the community by the user agents. This log is needed to compute *inDemand*, *selectivity*, and *satisfaction* metrics.
2. Response Time log tracks how long it took a community's Web service master to find and nominate a Web service slave to handle a user's request, the time needed by the Web service to provide the service back to the user, plus the extra time needed in case of substitution. This log is required to compute the *responsiveness* metric.
3. Advertized Response Time log tracks the community's advertized response time needed to provide the service back to the user. This log is used to compute the *satisfaction* metric in an objective way.
4. Invocations log tracks all received requests that have been accepted by the community master and invoked by the designated Web service slave. This log is required when computing the *fairness* metric.
5. Users log tracks the user agents that requested services from the community. This log contributes in the calculation of the *satisfaction* metric.

In addition to these logs, and to compute the provider-perceived performance metrics, provider agents should intercept the following additional time-stamped logs:

1. Subscriptions/Unsubscriptions logs track providers signing up/off to a community. These logs

are needed to compute the *market share* metric as they can tell how many Web service members are presently there in the community.

2. Leave log tracks Web services that leave the community. This log is used to compute the stability metric of the community.

The different logs permit to compute performance metrics in order to assess a community reputation. Data stored in these logs are monitored and collected by agents independent of the three key players in the computing platform (users, providers and community masters). This is aimed at fostering the trust between them.

4.2 Reputation-based operation

In the following, C_i denotes Community *i* that is under consideration by either provider or user, and $|C_i|$ denotes the number of Web services residing in C_i. We examine the role of a community reputation model from the perspectives of users of and providers of Web services.

Reputation from the user's perspective. As perceived by users, a community reputation model would help them select communities that can offer Web services that meet their quality expectations. The reputation model for communities would drive the discovery process that is based on aggregating and operating on historical performance metrics. In general, most of these metrics are monitored over an observation window, *w,* which is measured using different units like days or weeks, and recorded in run-time logs. Next, we outline some of the performance metrics that can play a role in assessing the reputation of a community as seen by users.

- *Responsiveness metric*: is one of the important qualities that users care about when selecting a community. It is the metric that determines how fast a community master is at nominating a Web service slave from this community that can handle the user's request, how fast the slave is at providing back the service, and how fast the substitution is performed. The responsiveness of community C_i, denoted by $Responsiveness_{C_i}$, is measured by computing the average of the *n* response times to the requests that took place throughout the observation window *w*. To simplify the formulas, the window *w*, which is supposed to be fixed, is omitted. So,

$$Responsiveness_{C_i} = \frac{1}{n}\sum_{k=1}^{n} rt_{C_i}^{k}$$

  where $rt_{C_i}^{k}$ is the response time taken to fulfill the user's request number *k* in C_i.

- *InDemand metric*: assesses the interest of users in a specific community by the percentage of requests that a community receives to the total requests submitted to all communities throughout *w*. That is,

$$InDemand_{C_i} = \frac{ReceivedRequests_{C_i}}{\sum_{k=1}^{M} ReceivedRequests_k}$$

  where *M* denotes the number of communities that are under consideration and $ReceivedRequests_{C_i}$ is the number of requests that C_i receives over *w*.

- *Satisfaction metric*: represents the objective opinion of clients who dealt with a community recently, i.e., over the observation window *w*. It is computed in terms of the difference between the advertized (promised) response time $art_{C_i}^{k}$ and the provided one $rt_{C_i}^{k}$ for each interaction *k* over *n* interactions during window *w*:

$$Satisfaction_{C_i} = e^{-\sum_{k=1}^{n} |rt_{C_i}^{k} - art_{C_i}^{k}|}$$

  Notice that with this function the satisfaction is 1 when the difference between the two times is null. The satisfaction decreases with the increase of this difference.

Reputation from the provider's perspective. A reputation model would here support providers of Web services identify the communities where their Web services will sign up. Below we list some performance metrics that we conceive important to the final assessment of the community's reputation.

- *Selectivity metric*: any provider wishes to join communities in which the provider has the maximum likelihood of being selected for fulfilling user's request compared to other communities. Assuming that the community master adopts fair (uniform) selection policy among Web services, the selectivity of community C_i is the average number of requests that are assigned to a Web service member inside a community and is computed as follows:

$$Selectivity_{C_i} = \frac{ReceivedRequests_{C_i}}{|C_i|}$$

- *MarketShare metric*: is the ratio of the Web services subscribed to a community to the total number of Web services subscribed to all communities:

$$MarketShare_{C_i} = \frac{|C_i|}{\sum_{k=1}^{M}|C_k|}$$

  where $\sum_{k=1}^{M}|C_k|$ represents the total number of Web services that signed up in all the M communities. From the provider's perspective, $MarketShare_{C_i}$ factor may be a double-edged sword. On the positive side, the larger $MarketShare_{C_i}$ the more powerful the candidate community is, which means a better chance of having it more often selected by clients. On the negative side, small $MarketShare_{C_i}$ means a lower selectivity rate and a tougher competition for the provider.
- Stability metric: is the metric allowing to measure how stable the community is in terms of the portion of Web services that leave the community during the window time w $Leave_{C_i}$ to the total number of members of this community:

$$Stability_{C_i} = e^{-\frac{Leave_{C_i}}{|C_i|}}$$

5. Conclusion

The notion of Web services communities is presented in this chapter where Web services providing the same functionality are put together in one or many communities. In these communities two behaviors emerge: intra and inter. The intra behavior is coopetative, which means Web services are supposed to be at the same time competitive since they offer users the same functionalities and cooperative since they serve the benefits of the same community. The inter behavior is a market-oriented behavior where communities operating in the same activity domains compete to attract the maximum number of users and providers of Web services. In such a competitive setting, providers of Web services use reputation metrics to decide about the community where to subscribe their Web services. Users use other reputation metrics to identify the community that offer the Web services they need.

Acknowledgments

The authors would like to thank Djamal Benslimane, Said Elnaffar, Chirine Ghedira, Michael Mrissa, Subramanian Sattanathan, Wei Wan, and Hamdi Yahyaoui, for their contributions to the early stages of the Web services community project. Jamal Bentahar would like to thank the Natural Sciences and Engineering Research Council of Canada (NSERC) and Fonds québécois de recherche sur la société et

la culture (FQRSC).

References

Benatallah, B., Sheng, Q. Z., & Dumas, M. (2003). The Self-Serv Environment for Web Services Composition. *IEEE Internet Computing*, 7(1).

Bengtsson, M. & Kock, S (2000). C*oopetition in Business Networks to Cooperate and Compete Simultaneously*. *Industrial Marketing Management*, 29(5).

Bentahar, J., Maamar, Z., Benslimane, D., & Thiran, Ph. (2007). An Argumentation Framework for Communities of Web Services. *IEEE Intelligent Systems*, 22(6), 75–83.

Bentahar, J., Maamar, Z., Wan, W., Benslimane, D., Thiran, P., & Subramanian, S. (2008). Agent-based Communities of Web Services: An Argumentation-driven Approach. *Service Oriented Computing and Applications*, 2(4), 219-238.

Bui, T., & Gacher, A. (2005). Web Services for Negotiation and Bargaining in Electronic Markets: Design Requirements and Implementation Framework. *Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS'2005), Big Islands, Hawaii, USA.*

Davies, J., Duke, A., & Sure, Y. (2003). OntoShare - A knowledge Management Environment for Virtual Communities. *Proceedings of the Second International Conference on Knowledge Capture (K-CAP'2003).*

Di Martino, B. (2009). Semantic Web Services Discovery based on Structural Ontology Matching, *International Journal of Web and Grid Services*, 5(1).

Elnaffar, S., Maamar, Z., Yahyaoui, H., Bentahar, J., & Thiran, Ph. (2008). Reputation of Communities of Web services - Preliminary Investigation. *Proceedings of the International Symposium on Web and Mobile Information Services (WAMIS'2008) held in conjunction of the 22nd International Conference on Advanced Information Networking and Applications (AINA'2008)*.

Khosravifar, B., & Bentahar, J., & Moazin, A,. & Thiran, Ph. (2010a). Analyzing Communities of Web Services Using Incentives, *International Journal of Web Services Research (IJWSR)*, 7(3).

Khosravifar, B., & Bentahar, J., & Moazin, A. (2010b). Analyzing the Relationships between some Parameters of Web Services Reputation. In *Proceedings of the 8th IEEE International Conference on Web Services (ICWS'2010)*, Miami, Florida, USA.

Maamar, Z., Benslimane, D., & Narendra, N. C. (2006). What Can Context do for Web Services? *Communications of the ACM*, 49(12).

Maamar, Z., Benslimane, D., Mostefaoui, G. M., Subramanian, S., & Mahmoud, Q. H. (2008). Toward Behavioral Web Services Using Policies, *IEEE Transactions on Systems, Man, and Cybernetics, Part A*, 38(6).

Maamar, Z., Subramanian, S., Thiran, P., Benslimane, D., & Bentahar, J. (2009). Communities of Web Services -Concepts, Architecture, Operation, and Deployment. International *Journal of E-Business Research*, 5(4), pp. 1-12. Hershey, PA: , IGI Global Publishing

Maaradji, A., Hacid, H., Daigremont, J., & Crespi, N. (2010). Towards a Social Network Based Approach for Services Composition, in *the Proceedings of the 2010 IEEE International Conference on Communications (ICC'2010)*, Cape Town, South Africa.

Maximilien, M., & Singh, M. (2002). Concept Model of Web Service Reputation. *SIGMOD Record*, 31(4).

Medjahed, B., & Atif, Y. (2007). Context-based Matching for Web Service Composition. *Distributed and Parallel Databases, Springer*, 21(1).

Medjahed, B., & Bouguettaya, A. (2005). A Dynamic Foundational Architecture for Semantic Web Services. *Distributed and Parallel Databases, Kluwer Academic Publishers*, 17(2).

Ouzzani, M., & Bouguettaya, A. (2004). Efficient Access to Web Services. *IEEE Internet Computing*, 8(2).

Schiaffino, S., & Amandi, A. (2004). User - Interface Agent Interaction: Personalization Issues. *International Journal of Human Computer Studies, Elsevier Sciences Publisher*, 60(1).

Smith, R. (1980). The Contract Net Protocol: High Level Communication and Control in Distributed Problem Solver. *IEEE Transactions on Computers*, 29.

Roson, R. (2005). Two-Sided Markets: A Tentative Survey. *Review of Network Economics*, 4(2).

Wan, W., & Bentahar, J., & Ben Hamza, A. (2010). Modeling and Verifying Agent-based

Communities of Web Services, *Proceedings of the 23rd International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA-AIE 2010)*, Cordoba, Spain.

Index words

- Web services communities
- Coopetition
- Cooperation
- Reputation
- QoS
- Contract Net Protocol
- Performance metrics
- Web services selection
- Web services substitution
- Web service providers attraction

Authors' bios and contact details

Zakaria Maamar is a full professor in the College of Information Technology at Zayed University, Dubai, United Arab Emirates. His research interests include Web services, social networks, and context-aware computing. He has a Ph.D. in computer science from Laval University, Quebec City, Canada.

Zakaria Maamar
Zayed University, PO BOX 19282, Dubai, United Arab Emirates
zakaria.maamar@zu.ac.ae

Philippe Thiran is an associate professor in the Faculty of Computer Science at the University of Namur, Belgium. He is also a member of the PReCISE Research center. His research interests include Web services, Adaptive information systems and Web engineering.

Philippe Thiran
University of Namur, 21, rue Grandgagnage, 5000 Namur, Belgium
pthiran@fundp.ac.be

Jamal Bentahar is an associate professor in the Concordia Institute for Information Systems Engineering at Concordia University, Montreal, Canada. His research interests include service engineering, multi-agent systems and formal verification. He has a Ph.D. in computer science and software engineering from Laval University, Quebec City, Canada.

Jamal Bentahar
Concordia University, 1515 Ste Catherine Street W,
EV 7.640, Montreal, Canada, H3G 2W1
bentahar@ciise.concordia.ca

[i] Web services that do not satisfy the QoS that a master Web service advertises and guarantees to potential users.

[ii] In case there are several tied bids, different selection opportunities are offered to the master Web service like randomly, firstly received, etc.