

Towards a Framework for Weaving Social Networks Principles into Web Services Discovery

Zakaria Maamar
Zayed University
Dubai, United Arab Emirates

Noura Faci
Université Lyon 1
Lyon, France

Youakim Badr
INSA-Lyon
Lyon, France

Leandro Krug Wives and
Pédro Bispo dos Santos
Universidade Federal do Rio
Grande do Sul
Porto Alegre, Brazil

Djamal Benslimane
Université Lyon 1
Lyon, France

José Palazzo Moreira de
Oliveira
Universidade Federal do Rio
Grande do Sul
Porto Alegre, Brazil

ABSTRACT

This paper presents a framework for weaving the principles (or metaphor) of social networks into Web services with emphasis on supporting their discovery. Despite the regular updates of the existing discovery techniques, the discovery of Web services continues to exacerbate users due to the large number of available Web services on the Internet. Different needs trigger the discovery of Web services such as developing new, value-added composite services and sustaining the high availability of other peers. These needs reveal three types of relationships between Web services, which are substitution, competition, and collaboration. The proposed framework includes tools that capture these relationships into social networks so that Web services discovery can be made smooth. Some implementation and simulation details are, also, discussed in this paper.

Categories and Subject Descriptors

H.4 [Information Systems Applications]: Miscellaneous;
D.2.11 [Software Architectures]: Service Oriented Architecture (SOA)

General Terms

Design

Keywords

Discovery, Social network, Web service

1. INTRODUCTION

Web services, which are considered as the flagship implementation technology of service-oriented architecture, fulfill

the promise of developing loosely-coupled, cross-enterprise business applications thanks to an exhaustive list of standards and specifications (WS-*) that service engineers can count on. Composition is another positive point in favor of Web services. Composition handles user's request that cannot be satisfied by any single, available Web service, whereas a composite Web service obtained by combining available Web services may be used.

With the tremendous popularity of Web applications and increasing reliance on mobile handheld devices, socializing over the Web has become an integral part of our daily lives. Looking for old friends and seeking recommendations on movies are some examples of how far Web socializing has gone. In a DEMOS (a London-based think tank focused on power and politics) report¹, enterprises were told to encourage their employees embrace, in a controlled way, social applications so they can establish and foster contacts with their colleagues, customers, and suppliers.

Different arguments are in favor of developing a framework that should help service engineers weave social networks' principles into Web services. They are dynamic by nature; they appear and disappear without prior notice. People do regularly the same by signing up/off with/from online applications. Web services, also, evolve in a changing environment in which they compete against other peers in order to take part in compositions [7], collaborate with other peers in order to answer complex users' needs [11], substitute other peers when these latter fail [15], just to cite few examples of all the interactions that Web services are engaged in and hence, can be used by service engineers to build these Web services' social networks. Although some researchers perceive Web services as passive components that react upon request only [6], others suggest that Web services can handle social elements [16, 21] in order to make Web services aware of their surroundings and past interactions. "Social" stems here from Web services interactions that correspond to aspects found in people's life like collaboration and competition.

In this paper, we address the specific issue of Web ser-

Permission to make digital or hard copies of all or part of this work for personal or classroom use is granted without fee provided that copies are not made or distributed for profit or commercial advantage and that copies bear this notice and the full citation on the first page. To copy otherwise, to republish, to post on servers or to redistribute to lists, requires prior specific permission and/or a fee.

WIMS '11, May 25-27, 2011 Sogndal, Norway

Copyright © 2011 ACM 978-1-4503-0148-0/11/05 ...\$10.00.

¹Bradwell, P. and Richard, R., "Network Citizens: Power and Responsibility at Work", Demos, 29 October 2008, www.demos.co.uk/publications/networkcitizens.

vices discovery using our social network-based framework. This discovery has been exacerbated by the increasing number of Web services that offer similar functionalities to users and other peers as well. Current discovery techniques are registry-based and rely on syntactic descriptions of these Web services' interfaces along with some additional semantic enrichments. Unfortunately syntactic discovery returns results with low precision and high recall ratios [12], and users' non-functional requirements are not taken into account during this discovery, which requires an additional step for identifying the appropriate Web services.

Our framework along with its different tools permits to reach two main goals: build Web services' social networks and capitalize on these networks to help Web Services discover other peers. On the one hand, building such networks require capturing over time the interactions that Web services engage in. Based on a previous work [14], we identify collaboration, substitution, and competition interactions that will be detailed later. On the other hand, using such networks requires navigating through their constituents for instance nodes and edges depending on the nature of the operation (e.g., collaboration, competition) to perform.

The remainder of this paper is organized as follows. Section 2 introduces briefly social networks and discusses the rationale of social Web services. Section 3 discusses the weaving of social networks into Web services. Sections 4, 5, and 6 present implementation details, related work, and concluding remarks, respectively.

2. BACKGROUND

In [13], we discuss the limitations associated with Web services including they know about themselves only, not about their users or peers; cannot reconcile ontologies among each other or with their users; cannot delegate their invocations; do not instantaneously and voluntarily cooperate with each other or self-organize; limit users' intervention considerably and operate as black boxes; and, consider only their own internal functional and non-functional details during execution and ignore other external details, such as user past interactions.

Assisting Web services socialize through appropriate tools means "showing" these Web services how to first, build their own networks of contacts and second, capitalize on these networks when they need to find peers with whom they like for example to collaborate. We identify two categories of social networks based on the intrinsic characteristics of Web services: similar and different. On the one hand, similarity means that Web services offer homogeneous functionalities (e.g., ticketBooking and ticketReservation are semantically equivalent) to the external community independently of their non-functional properties. We refine a similarity social network into two types: competition and substitution. On the other hand, different means that Web services offer heterogeneous functionalities (e.g., ticketBooking and weatherForecast) that can be put together for the benefit of the external community. We refine a difference social network into one type, only: collaboration (or composition). Competition, substitution, and collaboration are the three types of interactions that a Web service can engage in.

2.1 Social networks in brief

Simply put a social network is a graph that consists of nodes connected to each other through edges. The edges

are labeled with elements usually found in people's daily life like friendship, partnership, and dislike. The edges are sometimes directional, bidirectional, with weight, or a mixture of all of these. The size and shape of a social network vary over time for different reasons such as node deletion that affects outgoing and incoming edges.

To locate information in online social networks users navigate through them, screening the contents that their members post/refresh regularly. Some sites like LiveJournal and BlogSpot provide keyword-based search for textual or tagged content. Other sites like Youtube and GoogleVideo, also, propose a top-10 list of the most popular content, where the popularity is determined based on how often users screen the content or submit recommendations [3]. Social networking has been exploited to address other issues such as P2P resource management [8] and routing in ubiquitous computing environments [5].

2.2 Rationale of social Web services

Web services are basically software components deployed on the Internet. They are expected to function autonomously when they receive requests to satisfy. When this satisfaction requires several Web services at a time, we propose (in fact to service engineers) that the Web services use appropriate tools to keep track of (i) the peers with whom they have interacted, (ii) the outcomes of these interactions, (iii) the peers that have replaced them when they failed, (iv) why they were selected over other peers, etc. The variety and richness of interactions and cases allow the identification of at least three types of social networks that we gather into two main categories namely similarity and different as stated earlier.

A substitution social network is driven by the similarity of Web services' functionalities and stems out of the potential failures of Web services at run-time. To fix these failures, similar peers replace those that fail. In this social network the Web services substitute for each other. The more a peer is close (e.g., edge distance) to a Web service in the network, the more appropriate for substitution this peer is for this Web service.

A competition social network is, also, driven by the similarity of Web services' functionalities. Selecting appropriate Web services out of similar ones depends on users' non-functional requirements. In this social network, the Web services are in competition and just one Web service will be selected. The more a peer is close to a Web service in the network, the more competitiveness this peer is for this Web service.

A collaboration social network is driven by the difference in Web services' functionalities and stems out of users' requests that require the combination of several Web services. In this social network the Web services are ready to collaborate with each other. The more a peer is close to a Web service in the network, the more appropriate for collaboration this peer is for this Web service.

Fig. 1 shows the criteria that we use for building the above three social networks. These criteria are "non-functional properties" (QoS) and "semantic and policy matching". Matching refers to the efforts that are put into addressing semantic and policy conflicts. Semantic conflicts arise due to the potential mismatches between Web services' input and output data [10]. And policy conflicts arise due to the potential contradictions between Web services' internal regulations [19].

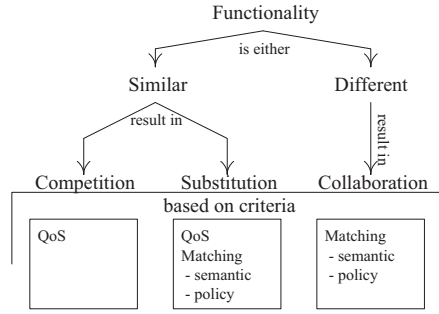


Figure 1: Criteria driving social networks building

Here we focus on semantic matching, only.

3. THE FRAMEWORK IN ACTION

Fig. 2 provides an overview of the framework for weaving social networks' principles into Web services discovery. Three levels are shown in this figure. The Web services level hosts different Web services made available for use. The tool level consists of a set of tools upon which service engineers rely to carry out this weaving. Last but not least, the social level stores the different social networks of Web services in a dedicated repository.

The weaving of social networks' principles into Web services discovery occurs through steps that are (Fig. 3): *identification of the components of a social network, matching analysis of Web services, social networks management, initial evaluation of the weights of edges, social networks navigation, ongoing evaluation of the weights of edges, and finally social networks ongoing management*. In the rest of this section the description of each step includes the necessary tools that services engineers use to complete the step.

3.1 Identification of the components of the social network step

The purpose of this step, which is linked to the configuration tool, is to work out the components upon which a social network is built. A network's nodes and edges correspond to Web services and interactions between Web services, respectively. We consider three types of interaction to support Web services discovery.

Collaboration interaction means that some Web services that participate already in ongoing compositions recommend to involve additional peers to be part of these compositions as well. For example, *AirBookingWS* recommends *TaxiBookingWS* to the user so that a taxi is secured at the destination city, although *TaxiBookingWS* was not initially requested when *AirBookingWS* was selected. Collaboration is subject to the complementarity of the functional properties between Web services, independently of the respective non-functional properties of these Web services².

Substitution interaction means that some Web services accept that similar peers replace them when they fail. For example *AirBookingWS* substitutes for *AirSeatReservationWS* if necessary. This kind of substitution is possible because of the similarity of the functional properties between the Web

services [15]. Plus this kind of substitution is quite different from the current substitution approaches that adopt copies of Web services to replace those that fail. To achieve a perfect substitution of Web services, the similarity of the non-functional properties of these Web services is highly recommended although this is not required (Fig. 1). The objective is to avoid conflicts between the failed Web service's pre- and post-peers and the substitute Web service.

Competition interaction means that some Web services have the same functional properties and just one Web service will be selected for invocation because of its capacity of satisfying the non-functional requirements of the user (Fig. 1). For example *AirBookingWS* and *AirSeatReservationWS* can be both used for the same purpose, which is airplane seat booking.

In Table 1, the **purpose** column highlights further the role of each social interaction in building Web services' social networks. The purpose of the collaboration interaction is to enrich compositions with additional Web services. The purpose of the substitution interaction is to make Web services highly-available. Finally, the purpose of the competition interaction is to develop compositions of Web services. This table also shows the impact of functional and non-functional properties on these interactions.

- In Row #1 - Table 1, "not-important" means that the non-functional properties do not really contribute to establishing the collaboration interaction; however the complementarity of the functional properties between Web services is critical.
- In Row #2 - Table 1, "similarity recommended" means that having similar non-functional properties on top of similar functional properties would simplify the substitution of Web services by avoiding conflicts between the new and existing Web services.
- In Row #3 - Table 1, "not important" means that the non-functional properties do not really contribute to establishing the competition relationship; however the similarity of the functional properties between Web services is critical.

3.2 Matching analysis of Web services step

The purpose of this step, which is linked to the matching tool, is to structure the interactions that may exist between Web services. This depends on the functional and non-functional properties (both forming a Web service's profile) of a Web service and calls for a matching analysis of these properties in term of either similarity or complementarity. The profile of a Web service is decomposed into five categories: Preconditions (P), Inputs (I), Outputs (O), Effects (E), and QoS.

Many approaches deal with Web services matching using their respective profiles. The matching consists of measuring the subsumption relationships between the concepts of each category in these profiles. For the matching tool in our framework, we adopt the work of Min et al. [17]. They use different matching algorithms to establish the degree of similarity between two Web services ws_i and ws_j . For each pair of concepts C_{ws_i} and C_{ws_j} in the same category (i.e., P/I/O/E/QoS), the Matching Score (MS) between these concepts is a function of (1) l , number of edges from C_{ws_i} to C_{ws_j} in the ontology, (2) h , depth of C_{ws_i} in

²In case of recommendation, the compatibility between the Web services' non-functional properties is assumed (Fig. 1).

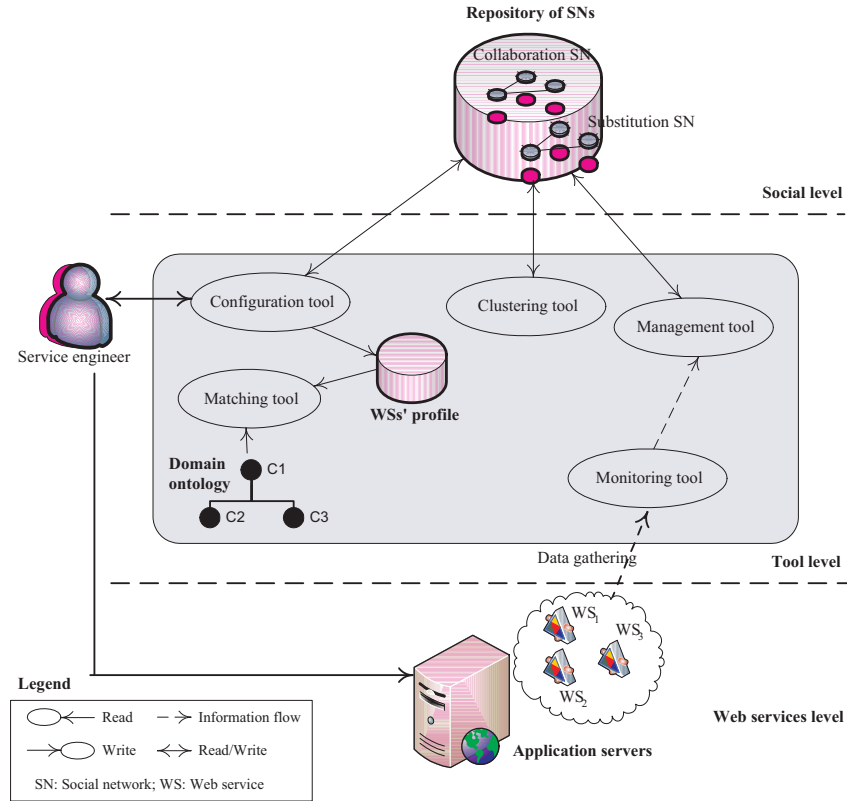


Figure 2: Framework general representation

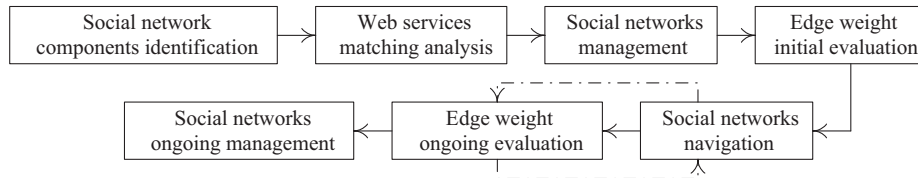


Figure 3: Proposed action plan

Table 1: Role of social interactions in Web services discovery

Type of social interaction	Purpose	Properties	
		functional	Non-functional
Collaboration	Composition enrichment	Complement	Not important
Substitution	High availability	Similar	Similarity recommended
Competition	Composition establishment	Similar	Not important

the ontology, and (3) d , local semantic density of C_{ws_i} and C_{ws_j} [23]. So, the matching score is computed as follows:

$$MS(C_{ws_i}, C_{ws_j}) = f_1 \times f_2 \times f_3 \quad (1)$$

where $f_1 = e^{\alpha l}$ with α as a constant, $f_2 = \frac{e^{\beta h} - e^{-\beta h}}{e^{\beta h} + e^{-\beta h}}$ with β as a smoothing factor, and $f_3 = \frac{e^{\lambda l} - e^{-\lambda l}}{e^{\lambda l} + e^{-\lambda l}}$ with λ as another smoothing factor.

The Degree of Similarity (DS) between ws_i and ws_j is calculated using the weighted average of the matching scores between the pairs of the profile types. Its value ranges between 0 and 1.

$$DS(ws_i, ws_j) = \frac{\sum_k w_k \times MS(C_{ws_{i_k}}, C_{ws_{j_k}})}{\sum_k w_k} \quad (2)$$

where k is the total number of concepts being similar and w_k is the weight associated with the matching score between a pair of concepts.

We adopt a similar approach to compute the Degree of Complementarity (DC) between ws_i and ws_j , which is denoted by $DC(ws_i, ws_j)$. We say that ws_i and ws_j are complementary when all the preconditions of ws_j match the effects of ws_i . So, we use Equation 2 for calculating the degree of complementarity, except that a matching score is performed between two concepts respectively in the precondition and effect categories.

Independently of the similarity/complementarity degree that might exist between Web services, these latter are connected in the future social networks through either (i) competition and substitution edges if their functional properties are declared similar (e.g., *AirBookingWS* and *AirSeatReservationWS*) or (ii) collaboration edges (e.g., *AirBookingWS* and *TaxiBookingWS*). It is worth noting that not all Web services have to collaborate with each other. Usually collaboration depends on factors such as (i) Web services belonging to the same domain for example travel (e.g., *AirBookingWS* and *TaxiBookingWS*) or (ii) data dependencies between Web services (e.g., *OutdoorActivityWS* depending on the feedback of *WeatherForecastWS*).

3.3 Social networks management step

The purpose of this step, which is linked to the management and clustering tools, is to either trigger the building of a new social network from scratch or extend an existing social network following the addition of new nodes and edges to this network. Node/edge deletion is handling in a different step (Section 3.7)

As per the interaction types that are now established between the Web services along with their respective degrees of similarity and complementarity, the management of the social networks in term of either construction or extension begins. To this end, we split the similarity degree value into “weak” (e.g., $0 \leq DS \leq 0.33$), “average” (e.g., $0.34 \leq DS \leq 0.66$), and “strong” (e.g., $0.67 \leq DS \leq 1$), and the complementarity degree value into “no” (e.g., $0 \leq DC \leq 0.49$) and “yes” (e.g., $0.5 \leq DC \leq 1$)³. These different degrees are later used to assign the initial values of the weights of edges that connect the Web services in the social networks. In the

³The initial values used for splitting similarity/complementarity degrees are chosen randomly. Other values can be used with no impact on social networks management.

rest of this paper, we focus on the degree of similarity as the same process applies to the degree of complementarity.

To simplify the construction of a social network for each Web service, we group Web services according to their degrees of similarity to a certain Web service into three clusters (Fig. 4 (a)), one cluster per range of degree of similarity. We refer to this Web service as **Web service root**. If a peer has a strong-similarity degree with the Web service root (e.g., $DS_{ws_i, ws_j} = 0.8$), then the management tool places this peer into the strong-similarity cluster of this root. A cluster might already be populated with other Web services. The placement of Web services into clusters is a continuous process that occurs as long as Web services are made available and accept to be discovered through social networks.

While the clustering of Web services is in progress, the connection of Web services together into the social networks is in progress as well, which leads into the extension of these social networks. Fig. 4 (b) illustrates the result of mapping the clusters onto the nodes and edges of a social network. The ways the initial values of these weights are obtained and the length of the edges between the Web services are both discussed in Section 3.4. In terms of numbers of edges, all Web services are of one edge only from the Web service root.

3.4 Initial evaluation of the weights of edges step

The purpose of this step, which is linked to the management tool, is to evaluate the weights of edges in a social network. As per Section 3.3, this evaluation takes place either during the construction of a new network or after adding a node to an existing network.

The initial value of the Weight of an Edge (WE) between ws_i and ws_j where ws_i is the Web service root corresponds to the degree of similarity between these Web services as per Equation 3. This initial value will change as soon as the discovery of Web services through the social networks becomes effective (Section 3.6). Each time a Web service is discovered through the social network of a peer, the weights of some edges are updated. Equation 3 calculates the initial weight of the edge connecting ws_i to ws_j at time 0 (t_0).

$$WE_{t_0}(ws_i, ws_j) = DS(ws_i, ws_j) \quad (3)$$

In Fig. 4 (b), the lengths of edges are different per type of cluster and for each Web service connected to the Web service root. Although multiple Web services are in the same cluster, the lengths of the respective edges that connect them to the Web service root are not the same. By doing this, the value of the degree of similarity is reinforced. We use the initial values of the edge weights to calculate the lengths of these edges as per Equation 4 where $Length(ws_i, ws_j)$ stands for the distance (e.g., in cm) of the edge connecting ws_i to ws_j , and n is any random positive number (e.g., 15) that is selected in this step. The purpose of n is to have representative distances for the edges since the weights’ values of these edges are small, i.e., between 0 and 1, and hence their graphical representation might not be clear.

$$Length_{t_0}(ws_i, ws_j) = n \times WE_{t_0}(ws_i, ws_j) \quad (4)$$

3.5 Social networks navigation step

The purpose of this step, which is linked to the management tool, is to develop mechanisms that allow to navigate

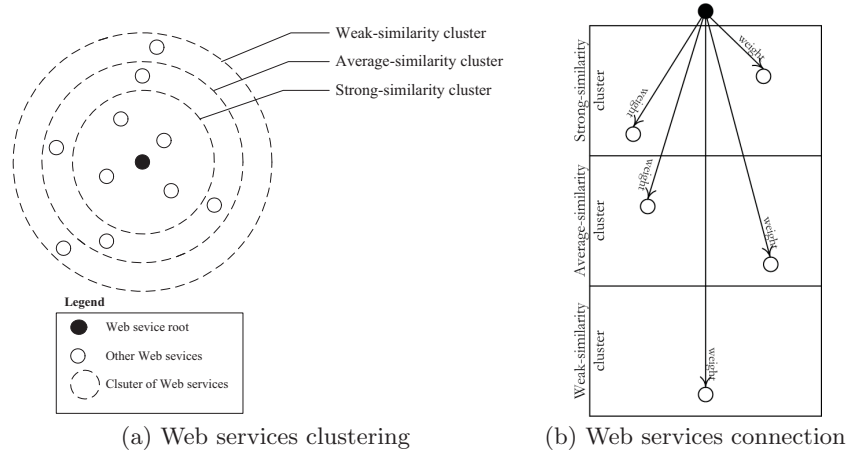


Figure 4: Social networks management

through the nodes and edges of a social network (Fig 4 (b)). Because of the collaboration, substitution, and competition relationships, the management tool suggests two options for deploying social networks based on their edges' weights.

In the first option, a Web service has three separate social networks, one per type of relationship that the Web service may have with other peers. If a Web service is not and will not be engaged in a certain relationship, the social network of this relationship will not be designed. In the second option, a Web service has one social network for the three relationships. To differentiate between these relationships in the social network, the weight of an edge is structured as a triple of values that correspond each to a specific relationship. If a value is set to *null* for example, this means that the Web service is not engaged in the relationship that corresponds to this value.

Independently of how the management tool structures the social networks, each Web service root constitutes an entry point to its social networks. As a result, the discovery of substitute, collaborative, or competitor peers starts by navigating through the different social networks commencing with the node of this Web service root. Depending on the activity that is ongoing whether collaboration, substitution, or competition, we identify the factors that drive the identification of the peers with respect to the needs of this activity as follows:

- P_c is the priority of screening a specific cluster of Web services (Fig. 4 (b)). Strong clusters have the highest priority, which means that the navigation of the social networks targets first the nodes and edges of these clusters. The priority of a cluster is fixed and thus, does not change over time.
- Co_{ws} is the cost that a Web service asks for when it is selected through a social network. The Web services located in the strong clusters have higher costs compared to those located in other clusters. This cost is proportional to the cluster's priority and inversely proportional to the weight of the edge that connects the Web service to the Web service root (Equation 5).

$$Co_{ws_j} = \frac{P_c}{1 + (P_c \times WE_{tn}(ws_i, ws_j))} \quad (5)$$

Since the weights of edges change as per Section 3.6, the cost of using a Web service is dynamic. Thus a Web service can be either promoted, which makes it move to a better cluster (e.g., from weak to average clusters), or demoted, which makes it move to a worst cluster (e.g., from strong to average clusters).

- E_{ws} represents previous experiences' satisfaction level when a Web service was selected through a social network and obviously deployed with success.
- L_{ws} represents the current load level of a Web service that needs to comply with its non-functional properties. The higher the level is, the less attractive the Web service is. Load assessment is linked to the monitoring tool.

We now define a selection function $Selection_{ws_j}$ which is an aggregation of these factors, as follows:

$$Selection_{ws_j} = \alpha_1 \times Co_{ws_j} + \alpha_2 \times E_{ws_j} + \alpha_3 \times (1 - L_{ws_j}) \quad (6)$$

where $\sum_{i=1,3} \alpha_i = 1$.

After applying Equation 6 to the Web services in a social network, the management tool returns the peer with the highest $Selection_{ws}$'s value as it is the most appropriate substitute, collaborative, or competitor peer.

3.6 Ongoing evaluation of the weights of edges step

The purpose of this step, which is linked to the management tool, is to reinforce the role of social networks in Web services discovery. This reinforcement happens by updating the weights of edges each time a substitute, collaborative, or competitor peer to a Web service is discovered thanks to these social networks. We recall that the initial values of weights of edges are obtained using the degrees of similarity (Equation 3).

To update the weights of edges, appropriate functions are to be suggested per type of relationship. For example, the function to evaluate the weights in a substitution social-network can be given by the formula used in reward-based price mechanisms [24] (Equation 7).

$$WE_{t+\delta t}(ws_i, ws_j) = WE_t(ws_i, ws_j) + \alpha \times \left(\frac{|ws_j \text{ selection}_{t+\delta t}|}{|ws_i \text{ failure}_{t+\delta t}|} - WE_t(ws_i, ws_j) \right) \quad (7)$$

where α is a constant between 0 and 1, δt represents the update period, $|ws_j \text{ selection}_T|$ is the number of times that ws_j has substituted for ws_i following the use of ws_i 's substitution social network at time T, and $|ws_i \text{ failure}_T|$ is the number of times that ws_i failed at time T. In Equation 7, it is expected that ws_j did not fail as well so that the update of weights is finalized.

Because of the changes in the weights' values affecting the distances of edges that connect the Web services to the Web service root in a social network (Equation 8), this results in either promoting or demoting a Web service from a cluster to another.

$$Length_{t+\delta t}(ws_i, ws_j) = WE_{t+\delta t}(ws_i, ws_j) \times n \quad (8)$$

As shown in Fig. 3, the actions of the steps discussed in Section 3.5 and Section 3.6 form a cycle; they are executed back and forth, e.g. Step 5 $\rightarrow$ Step 6 $\rightarrow$ Step 5 $\rightarrow$ Step 6 $\rightarrow \dots$.

3.7 Social networks ongoing management step

The purpose of this step, which is linked to the management tool again, is to review the shape of a social network following the occurrence of different events like edge's weight dropping below a certain threshold, Web services ceasing to exist, etc. All these events require deleting nodes and edges (incoming and outgoing), which impacts the shape of a social network.

4. IMPLEMENTATION

The implementation of our framework to weave social networks' principles into Web services consists of two steps. The first step is similarity assessment through the matching tool and the second step is substitution analysis through the management tool.

In the first step, we selected *Service51.Specialist* and *Service12.Miscellaneous* (*Service51* and *Service12* for short), from a set available at andreas-hess.info/projects/annotator/index.html. Both belong to the dictionary category whose concepts are illustrated in Fig. 5. *Service51* translates English word into Pig Latin, and *Service12* translates a Word from one language into another. Only input and output elements are used in building their respective profiles (Fig. 6).

The matching tool uses Equation 2 to assess the degree of similarities between two Web services. The values of α and β were assigned to 1, l is the number of arrows for reaching a common superclass between the concepts being analyzed, and h is the depth of the concept in the ontology taxonomy, being 0 the depth of owl:Thing. The weights for the pairs formed by the Input and Output categories were the same.

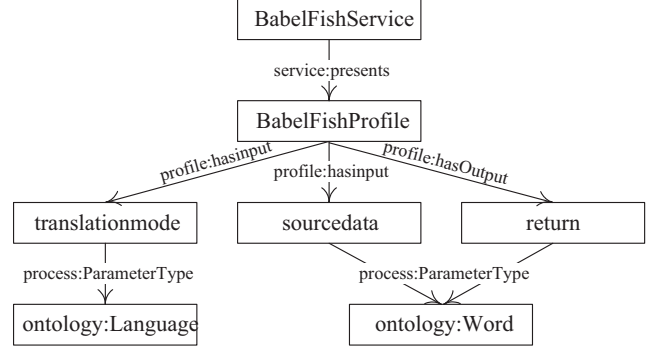


Figure 6: *Service12*'s profile

The similarity between *Service51* and *Service12* is calculated as follows:

- The pair of concepts formed by the Input category from *Service51* and *Service12* is: $\{(Word, Word), (Word, Language)\}$.
- The pair of concepts formed by the Output category from *Service51* and *Service12* is: $\{(Word, Word)\}$.

- The matching scores for these pairs are:

$$MS(Word, Word) = e^0 \times \left(\frac{e^2 - e^{-2}}{e^2 + e} \right) = 0.964$$

$$MS(Word, Language) = e^{-1} \times \left(\frac{e^2 - e^{-2}}{e^2 + e} \right) = 0.355$$

- Having the matching score among the pair of concepts of each category computed, the similarity degree between the two Web services is:

$$DS(Service51, Service12) = \frac{MS(Word, Word) + MS(Word, Word) + MS(Word, Language)}{3} = \frac{0.964 + 0.964 + 0.355}{3} = 0.761$$

In the second part, we worked on substitution by failing *Service12*. To this end the strong, average, and weak clusters were built (Figure 4 (a)). We examined the impact of a Web service's load (L_{ws}) on accepting substitution requests. Different scenarios were planned as follows. Fig. 7 is a partial listing of the experimental values used in these scenarios (S and W refer to Selection and Weight as per Equations 6 and 7, respectively).

- Scenario 1: Each Web service's load is set to null ($L = 0$). Amongst the Web services in the strong cluster, the management tool selects *Service51* ($W = 0.76$ and $S = 2.50$).
- Scenario 2: *Service51* becomes fully loaded ($L = 1$). *Service52* that is in the same cluster as *Service51* is selected ($W = 0.96$ and $S = 2.45$) instead of *service51* ($W = 0.77$ and $S = 1.49$).
- Scenario 3: *Service51* and *Service52* are now fully loaded. *Service95* in the strong cluster is selected.
- Scenario 4: All Web services in the strong cluster become fully loaded after a couple of iterations. *Service53* in the average cluster is selected ($W = 0.36$ and $S = 2.42$).



Figure 5: Taxonomy of concepts used during implementation

Network weights for Service12

List of available Web services	Iteration 0							Iteration 1							Iteration 2							
	W	Cluster	Co	E	L	S	subs	W	Cluster	Co	E	L	S	subs	W	Cluster	Co	E	L	S	subs	
1 service2	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
2 service6	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
3 service12	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
4 service17	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
5 service20	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
6 service22	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
7 service30	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
8 service38	0.56	Average	0.39	1.00	0.00	2.39	0	0.56	Average	0.39	1.00	0.00	2.39	0	0.56	Average	0.39	1.00	0.00	2.39	0	
9 service51	0.76	Strong	0.50	1.00	0.00	2.50	1	0.77	Strong	0.49	1.00	1.00	1.49	1	0.77	Strong	0.49	1.00	1.00	1.49	1	
10 service52	0.96	Strong	0.45	1.00	0.00	2.45	0	0.96	Strong	0.45	1.00	0.00	2.45	1	0.94	Strong	0.46	1.00	1.00	1.46	1	
11 service53	0.36	Average	0.42	1.00	0.00	2.42	0	0.36	Average	0.42	1.00	0.00	2.42	0	0.36	Average	0.42	1.00	0.00	2.42	0	
12 service60	0.66	Average	0.38	1.00	0.00	2.38	0	0.66	Average	0.38	1.00	0.00	2.38	0	0.66	Average	0.38	1.00	0.00	2.38	0	
13 service76	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
14 service85	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
15 service91	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	0.13	Weak	0.19	1.00	0.00	2.19	0	
16 service95	0.96	Strong	0.45	1.00	0.00	2.45	0	0.96	Strong	0.45	1.00	0.00	2.45	0	0.96	Strong	0.45	1.00	0.00	2.45	1	
Scenario 1							Scenario 2							Scenario 3								
Strongest: 0.96							Strongest: 0.96							Strongest: 0.96								
Subs. Weight 2.50							Subs. Weight 2.45							Subs. Weight 2.45								
Position: 9							Position: 10							Position: 16								
Element: service51							Element: service52							Element: service95								
cluster amount							cluster amount							cluster amount								
Strong 3							Strong 3							Strong 3								
Average 3							Average 3							Average 3								
Weak 10							Weak 10							Weak 10								
16							16							16								

Figure 7: Some experimental values during simulation

- Scenario 5: We keep the previous experimental conditions unchanged, except for *Service53* whose load value is now set to 0.5. *Service60* in the average cluster is selected.
- Scenario 6: All Web services in the strong and average clusters become either fully loaded ($L = 1$) or half loaded ($L = 0.5$). *Service2* in the weak cluster is selected.

We moved afterwards to study the impact of Web services' promotion and demotion in a social network (Fig. 4 (b)). This happens based on user satisfaction rating. The following scenarios were established using the aforementioned Scenario 6's experimental conditions:

- Scenario 1: Several iterations were required before observing the transition of *Service2* from weak to average clusters. As the weight of *Service2* increases through the iterations, it was selected several times.
- Scenario 2: *Service2*'s load was changed from zero ($L = 0$) to half ($L = 0.5$). *Service2* continued to be selected.
- Scenario 3: Each Web service becomes idle ($L = 0$). *Service51* in the strong cluster is selected again.
- Scenario 4: In the following iteration, we assumed that *Service51* failed in satisfying the substitution request ($E = 0$). In this case, *Service95* in the same cluster is selected.
- Scenario 5: When we reiterated the selection process, we noted that *Service95* moves from strong to average clusters as its cost ($Co_{Service95}$) calculated as per Equation 5 decreases.

5. RELATED WORK

Major research challenges in Web services can be split into two main streams: how to find Web services with respect to the descriptions that users provide of their needs

and how to differentiate between Web services with similar functionalities using QoS properties. Existing standards for Web services discovery such as UDDI let organizations publish their Web services and allow keyword-based search of these Web services using tModel elements. As UDDIs proliferate along with their size increase, screening multiple UDDIs has become time consuming and ineffective [1]. Alternative discovery approaches use Web services' operations. In [18], Sajjanhar et al. propose information retrieval techniques such as Term Frequency and Inverse Document Frequency to identify a Web service's operations that are almost similar to a user description. These techniques consider all textual elements in a WSDL document in order to extract and compare them against the textual elements of the available Web services. Unfortunately, these techniques can not reveal the semantic relationship between Web services. Matching algorithms deal with Web services discovery in a different way [22]. Given a textual description of a Web service, these algorithms retrieve the desired Web services that match operation structures. The drawbacks of these algorithms stem from the fact that simple structural matching may be invalid when two Web services' operations have many similar data type substructures.

In the semantic Web community, Description Logic is used to develop inferring mechanisms to establish similarity between Web services and a given Web service [20]. In addition, ontology mapping [9] and semantic processing are exploited widely to annotate Web services (e.g., SAWSDL, DAML-S, or OWL-S) [4]. Most of these approaches depend on the names of parameters and operations and do not deal with the QoS properties. Web services discovery based on QoS search criteria has been studied extensively to retrieve the most relevant Web services. Ontology matching, clustering, and similarities of metric values [2] just to mention a few, are fields that the service-oriented community examines. The main lack of all these representative approaches is the discovery of Web services independently from each other. In our work, we first discover Web services according to a specific context that could be collaboration, compe-

tion, or substitution and second, pay a special attention to Web services' histories, past compositions, and dynamic relations. Extending the social network paradigm to contextual Web service discovery is particularly useful to find most relevant Web services, which not only provide the right functional and non-functional properties but, also, the appropriate Web services for a specific context driven by the needs of collaboration, competition, or substitution. The power of social networks stems from their ability to capture real-world phenomena such as collaboration, competition, and partnership.

6. CONCLUSION

In this paper, we presented our framework to weave social networks' principles into Web services discovery. Different steps along with different tools were identified including identification of the components of a social network, matching analysis of Web services, management of the social networks, initial evaluation of the weights of edges of these social networks, navigation through these social networks, ongoing evaluation of the weights of these edges, and, finally, ongoing management of these social networks. We adopted social networks' principles because they capture the relationships that exist between Web services. Collaboration, substitution, and competition are examples of these relationships. Each relationship is subject to analyzing the similarity and complementarity of the functionalities of Web services. We clustered the Web services depending on the outcome of this analysis prior to mapping these clusters onto social networks. Weaving social elements into Web service operation means **Social Web Services** that will establish and maintain networks of contacts; put their users in the heart of their life cycle enabling additional functionalities through, e.g., collaboration and annotation, count on their (privileged) contacts when needed; form with other peers strong and long lasting collaborative groups; and, know with whom to partner so ontology reconciliation is minimized [13]. Our future work consists of fine tuning the implementation and comparing for example discovery time using our social networks and other registry-based means.

Acknowledgments

Z. Maamar's work was partially supported by the ZU Research Incentive Funds Program (Grant #R09015). L. Krug Wives's, J. P. Moreira de Oliveira's, and P. B. dos Santos's work was partially supported by CNPq - Conselho Nacional de Desenvolvimento Científico e Tecnológico, Brazil.

7. REFERENCES

- [1] E. Al-Masri and Q. H. Mahmoud. WSCE: A Crawler Engine for Large-Scale Discovery of Web Services. In *Proceedings of the IEEE International Conference on Web Services (ICWS'2007)*, Salt Lake City, Utah, USA.
- [2] E. Al-Masri and Q. H. Mahmoud. Discovering the Best Web Service. In *Proceedings of the 16th International Conference on World Wide Web (WWW'2007)*, Banff, Alberta, Canada, 2007.
- [3] S. Amer Yahia, M. Benedikt, and L. Lakshmanan. Efficient NetworkAware Search in Collaborative Tagging Sites. In *Proceedings of the 34th International Conference on Very Large Databases (VLDB'2008)*, Auckland, New Zealand, 2008.
- [4] S. Bansal and J. Vidal. Matchmaking of Web Services based on the DAML-S Service Model. In *Proceedings of the Second International Joint Conference on Autonomous Agents and Multiagent Systems (AAMAS'2003)*, 2003.
- [5] G. Bigwood, D. Rehunathan, M. Bateman, T. Henderson, and S. Bhatti. Exploiting Self-Reported Social Networks for Routing in Ubiquitous Computing Environments. In *Proceedings of the 1st International Workshop on Social Aspects of Ubiquitous Computing Environments (SAUCE'2008) held in conjunction with the 4th IEEE International Conference on Wireless and Mobile Computing, Networking, and Communications (WiMob'2008)*, Avignon, France, 2008.
- [6] K. P. Birman. Like It or Not, Web Services Are Distributed Objects. *Communications of the ACM*, 47(12), December 2004.
- [7] T. Bui and A. Gacher. Web Services for Negotiation and Bargaining in Electronic Markets: Design Requirements and Implementation Framework. In *Proceedings of the 38th Hawaii International Conference on System Sciences (HICSS'2005)*, Big Island, Hawaii, USA, 2005.
- [8] V. Carchioli, M. Malgeri, G. Mangioni, and V. Nicosia. Exploiting Social Networks Dynamics for P2P Resource Organisation. In *Proceedings of the 21st International Symposium on Computer and Information Sciences (ISCIS'2006)*, Istanbul, Turkey, 2006.
- [9] S. Chaari, Y. Badr, and F. Biennier. Enhancing Web Service Selection by QoS-based Ontology and WS-Policy. In *Proceedings of the 2008 ACM symposium on Applied computing (SAC'2008)*, 2008.
- [10] B. Di Martino. Semantic Web Services Discovery based on Structural Ontology Matching. *International Journal of Web and Grid Services*, 5(1), 2009.
- [11] M. Fluegge, I. J. G. Santos, N. Paiva Tizzo, and E. R. M. Madeira. Challenges and Techniques on the Road to Dynamically Compose Web Services. In *Proceedings of the 6th International Conference on Web Engineering (ICWE'2006)*, Palo Alto, California, USA, 2006.
- [12] K. Kritikos and D. Plexousakis. Semantic QoS Metric Matching. In *Proceedings of the 4th IEEE European Conference on Web Services (ECOWS'2006)*, Zurich, Switzerland, 2006.
- [13] Z. Maamar, H. Hacid, and M. N. Hunhs. Why Web Services Need Social Networks. *IEEE Internet Computing*, 2011 (forthcoming).
- [14] Z. Maamar, L. Krug Wives, Y. Badr, and S. Elnaffar. Even Web Services Can Socialize: A New Service-Oriented Social Networking Model. In *Proceedings of the International Conference on Intelligent Networking and Collaborative Systems (INCoS'2009)*, Madrid, Spain, 2009.
- [15] Z. Maamar, Q. Z. Sheng, S. Tata, D. Benslimane, and M. Sellami. Towards an Approach to Sustain Web Services High-Availability Using Communities of Web Services. *International Journal of Web Information*

Systems, 5(1), 2009.

- [16] A. Maaradji, H. Hacid, J. Daigremont, and N. Crespi. Towards a Social Network Based Approach for Services Composition. In *Proceedings of the 2010 IEEE International Conference on Communications (ICC'2010)*, 2010.
- [17] L. Min, S. Weiming, H. Qi, and Y. Junwei. A Weighted Ontology-based Semantic Similarity Algorithm for Web Services. *Expert Systems with Applications*, 36(10), December 2009.
- [18] A. Sajjanhar, J. Hou, and Y. Zhang. Algorithm for web services matching. *Lecture notes in computer science*, 3007:665–670, 2004.
- [19] Q. Z. Sheng, J. Yu, Z. Maamar, W. Jiang, and X. Li. Compatibility Checking of Heterogeneous Web Service Policies Using VDM++. In *Proceedings of the IEEE Workshop on Software and Services Maintenance and Management (SSMM'2009) held in conjunction the 2009 IEEE Congress on Services, Part I (SERVICES I'2009)*, 2009.
- [20] A. Sudhir and S. Rudi. Automatic Matchmaking of Web Services. In *Proceedings of the IEEE International Conference on Web Services (ICWS'2006)*, Washington, DC, USA, 2006.
- [21] X. Xie, B. Du, and Z. Zhang. Semantic Service Composition based on Social Network. In *Proceedings of the 17th International World Wide Web Conference (WWW'2008)*, Beijing, China, 2008.
- [22] Y. Yanan and Z. Yanchun. Web Services Discovery based on Schema Matching. In *Proceedings of the Thirtieth Australasian Conference on Computer Science (ACSC'2007)*, Ballarat, Victoria, Australia, 2007.
- [23] B. Yhuha Li, Zuhair A., and D. McLean. An Approach for Measuring Semantic Similarity between Words Using Multiple Information Sources. *IEEE Transactions on Knowledge and Data Engineering*, 15(4), 2003.
- [24] B. Yu, Cuihong Li, M.P. Singh, and K. Sycara. A dynamic pricing mechanism for p2p referral systems. In *AAMAS'04: International Conference on Agents and Multi-Agent Systems*, New York, USA, 2004.