Towards a User-Centric Social Approach to Web Services Composition, Execution, and Monitoring

Zakaria Maamar[1], Noura Faci[2], Quan Z. Sheng[3], and Lina Yao[3]

[1] Zayed University, Dubai, U.A.E
[2] Université Lyon 1, Lyon, France
[3] The University of Adelaide, Adelaide, Australia

Abstract. This paper discusses the intertwine of social networks of users and social networks of Web services to compose, execute, and monitor Web services. Each network provides details that permit achieving this intertwine and thus, completing the three operations. A user social-network is used to advise users on the next Web services to select based on their peers' experiences, whereas a Web service social network is used to advise users on the substitutes to select in case a Web service fails, for example. To make the intertwine of these social networks happen, three components are developed: composer, executor, and monitor. The social composer develops composite Web services considering relations between users and the ones between Web services. The social executor assesses the impact of these relations on these composite Web services execution progress. Finally, the social monitor replaces failing Web services to guarantee the execution continuity of these composite Web services. A running example and a prototype illustrate and demonstrate the intertwine of these social networks, respectively.

Keywords: Web service, service composition, social network.

1 Introduction

Over the years, different development waves have shaped the Web. Started as a simple browsing tool to screen Web sites, the Web now is a dynamic and robust platform upon which organizations conduct business and people engage in cross-organization collaborative activities. The latest development wave in the Web triggered by among other things the pressure on organizations to remain agile and Web 2.0 widespread adoption, sheds the light on two major research streams:

- Research on loosely-coupled business applications. The execution of these applications spans several distributed and heterogeneous systems and hence, has to cross organization boundaries transparently. Service-Oriented Architecture (SOA) and its flagship implementation technology known as Web services [15] are a response to the challenges that this type of execution poses on organizations.
- Research on social computing illustrated with the massive deployment of social applications like Facebook and LinkedIn. These applications capitalize on the ability and willingness of users to interact, share, collaborate, and recommend. Users are nowadays referred to as *prosumers*, i.e., providers and consumers at the same

X.S. Wang et al. (Eds.): WISE 2012, LNCS 7651, pp. 72–86, 2012.

time [16,21]. However, the richness and complexity of information in these applications pose challenges on how to capture and structure these information for future use while preserving users' privacy and information sensitivity.

Although the aforementioned research streams are pursued separately, they share a common element, the Web, as an execution platform for cross-organization processes and an exposure means for organizations. It would be tempting to examine why and how both streams can be blended (i.e., interleave their use) together as this might yield interesting results for instance, developing business applications that consider social elements (e.g., users' past experiences) in their operation. However, the success of this blend is subject to addressing questions like how to advise users on the necessary building blocks (e.g., Web services) that they need to use for developing such business applications, what building blocks to select as per these users' needs, how to make sure that conflicts will not raise when separate building blocks are put together, and how to capture the interactions between the building blocks for better use in the future.

Our literature review identified two major but independent research initiatives that look into Web services-based business applications from a *social* perspective. In the first initiative illustrated by [10,20], users are in the center of developing complex, value-added composite Web services. Capitalizing on social networks of users, users are advised on the next step to take based on their social "entourage". In the second initiative [5,7], Web services are in the center of discovering peers when developing complex, value-added composite Web services. Capitalizing on social networks of Web services, Web services are labeled as collaborators, substitutes, or competitors. In this paper, we explore the blend of these two initiatives, i.e., social networks of users and social networks of Web services, to develop a user-centric social approach to Web services composition, execution, and monitoring. The objective is to assess and illustrate the value-added of these networks to the cycle of composition, execution, and monitoring. Composition means making Web services take part in composite Web services based on existing social relations between users and between Web services as well. Execution means triggering Web services with respect to these social relations. Finally, monitoring means keeping Web services on alert in case changes in these social relations happen so that actions are taken.

Section 2 discusses the rationale of interleaving social networks of users and of Web services and suggests a literature review on how these networks permit developing business applications. Section 3 details our user-centric social approach to compose, execute, and monitor Web services. A prototype system implementation is reported in Section 4 before concluding and identifying some future work elements in Section 5.

2 Background

2.1 Rationale of Social Networks Interleaving

As stated in Section 1, interleaving the use of social networks of users and social networks of Web services might yield interesting results for developing Web services-based business applications. We discuss hereafter the motivations of this interleaving and the requirements to satisfy during the interleaving.

On the one hand, social networks of users record interaction experiences of users with Web services over time so that these experiences are captured and shared later with other peers. Assuming that users' feedbacks on these interactions are fair (i.e., unbiased), it becomes possible to advise users on where to look for Web services, how to select Web services, and what to expect out of Web services. Web services' non-functional properties [11] (e.g., response time and execution time) do not include such details, so limited advice is provided. This shows the value-added of social networks of users to the cycle of composing, executing, and monitoring Web services.

On the other hand, social networks of Web services record the situations that Web services come across at run time [5]. These situations known as collaboration, competition, and substitution permit to advise users on which Web services can or like to collaborate with each other, which Web services can be selected over a peer, and which Web services can replace a failing peer. Similarly, Web services' non-functional properties do not include such details, so limited advice is provided. This shows the value-added of social networks of Web services to the cycle of composing, executing, and monitoring Web services. The management of social networks of Web services in terms of creation, access, and maintenance is discussed in [8].

Interleaving the use of these two types of social networks needs to take into account the requirements that are posed on each step of the aforementioned cycle. In particular,

- Requirements on composition refer to Web services discovery and selection, i.e., looking for the necessary Web services while taking into account users' needs, users' social relations, and Web services' social relations.
- Requirements on execution refer to satisfying the requirements posed on compositions at run-time, i.e., assessing the impact of considering users' social relations and Web services' social relations as well on the progress of these compositions.
- Requirements on monitoring refer to the continuity of Web services execution when failures arise, i.e., looking for peers that can substitute for the failing Web services while taking into account the social relations of these failing Web services.

2.2 Literature Review

Our work is at the cross-road of two main research streams: social computing (exemplified with Web 2.0) and service-oriented computing (exemplified with Web services). Existing works either adopt Web services to support social networks of users, or develop social networks of Web services to support users identify collaborator, substitute, and competitor Web services. The combination of both streams is quite new and several research opportunities are still un-taped. To the best of our knowledge, this work is the first attempt to examine such a combination. Existing works adopt either social networks of users or social networks of Web services to build composite Web services. The combination of these networks is totally absent.

In the category of social networks of users, Maaradji et al. propose a social composer (*SoCo*) that advises users on the next actions to take in response to specific events like selecting certain Web services [9]. Xie et al. introduce a framework for semantic service composition based on social networks [20]. Wu et al. rank Web services using non-functional properties and invocation requests at run-time [19]. A Web service's

popularity as analyzed by users is the social element used during ranking. Tan et al. apply social networks analysis to mine and analyze a workflow repository, focusing on service usage patterns [18]. Nam Ko et al. discuss the social Web in which a new type of services called "social-networks connect services" help third party develop social applications without having them build social networks [13]. Last but not least, Al-Sharawneh and Williams mix semantic Web, social networks, and recommender systems to assist users in selecting Web services with respect to their functional and non-functional requirements [1]. Besides the "market-leader" concept that refers to the best Web service, Al-Sharawneh and Williams develop two ontologies called "follow-leader" to classify users and "preference" to specify users' preferences.

In the category of social networks of Web services, we cite our research works in [4] and [7]. In the first work, we suggest a method to engineer "social Web services". Questions addressed in this method include what relations exist between Web services, what social networks correspond to these relations, how to build social networks of Web services, and what social behaviors can Web services exhibit. In the second work we use social networks to support Web services discovery. Different social networks permit to describe the situations that Web services encounter for instance collaboration and recommendation. These situations mean that Web services are not isolated components that respond to user requests, only. Contrarily, Web services compete against other similar peers during selection, collaborate with other different peers during composition, and may replace other similar peers during execution despite the competition[1].

3 Proposed Approach

3.1 Overview and Illustration

Our approach to interleave the use of social networks of users and social networks of Web services is built upon *social composer*, *social executor*, and *social monitor* components. The role and duties of each component are described hereafter briefly. Fig. 1 shows these components along with a repository of social networks of users and social networks of Web services, a pool of users who will populate the social networks of users, and a pool of Web services that will populate the future compositions.

The social composer relies on the social networks of users and social networks of Web services to advise users on how to build composite Web services. Examples of advice concern (i) which Web services to include in these compositions [9], (ii) which Web services to check in case some decline to participate in these compositions [6], and (iii) which Web services to select to ensure a better compatibility level of these compositions despite their loose-coupling nature [17].

The social executor does not provide advice on compositions. Instead it assesses the impact of the social composer's advice (when considered) on the execution progress of compositions. The social executor feeds the social composer with details so that the social composer updates the necessary social networks. These details include (i) how the

[1] Simultaneous competition and substitution, *a.k.a* coopetition [2], refers to Web services that compete to take part in compositions and also collaborate to support each other during failure.

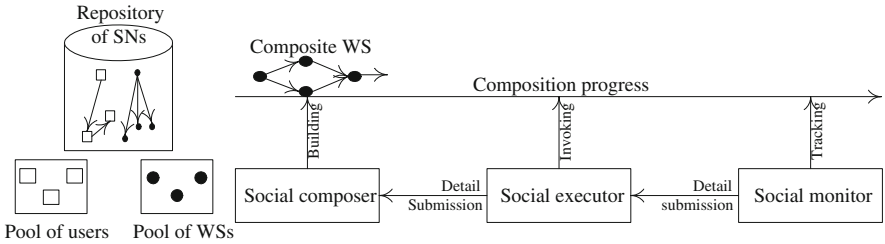


Fig. 1. Composer, executor, and monitor social components in action

Web services that are suggested through the social networks performed and (ii) which Web services that are also suggested did not join the compositions.

The social monitor relies on the social networks of Web services to advise users on which Web services to check in case those that are already taking part in some ongoing compositions fail. The social monitor feeds the social executor with details so that this latter updates the social networks of Web services for the benefit of the social composer. These details include (i) which Web services failed, (ii) which Web services replaced those that failed, (iii) how the replacing Web services performed, and (iv) how the Web services that are already in compositions reacted to the replacing Web services. Out of these details, the social monitor does more than a simple monitoring but puts forward different solutions for the social composer like assessing Web services performance.

To show how the composer, executor, and monitor components support the interleaving of social networks of users and social networks of Web services happens, we suggest the following scenario. Needless to mention the simplifications made in the scenario for the sake of explanation. A businesswoman who has got a 3-day stop over in a city decides to visit some museums among other sightseeing activities. She logs into a Web site and invokes `museumVisitWS` submitting her preferences and constraints. Different cases are listed hereafter to illustrate the role of each component in Fig. 1.

1. Prior to executing `museumVisitWS`, the social composer consults the businesswoman's social networks, finding out that some friends who visited the city before recommend riding taxis at this time of the year due to unexpected heavy rains.
2. To identify a Web service for taxi booking, the social composer consults `museumVisitWS`'s social networks to find out that `museumVisitWS` has frequently and successfully collaborated with `taxiBookingWS`, which is subsequently selected to arrange taxi booking. Another Web service called `translatorServiceWS` is also advised by the social composer as reported in the social networks of `museumVisitWS`, but this time our businesswoman declines the advice since she is familiar with the language spoken in the city.
3. When the selection of Web services is complete, the social executor invokes them while keeping an eye on all the Web services that are added to the composition through the social networks of users and social networks of Web services. The objective is to reflect the performance of these Web services on the different networks.

4. At run time, `museumVisitWS` fails just after `taxiBookingWS` is done. The social monitor takes immediate actions by consulting `museumVisitWS`'s social networks and recommends `museumTourWS` instead, as a substitute.

The aforementioned cases show some of the advantages of the social networks to the cycle of Web services composition, execution, and monitoring. It is for sure that some of these cases can be handled by screening registries, but Web services' previous experiences and users' advice are not captured and hence, overlooked during this screening.

3.2 Laying Down the Foundations

Our user-centric social approach revolves around social networks of users as per Maaradji et al. [9] and social networks of Web services as per Maamar et al. [7]. We interleave these network types as per the requirements listed in Section 2.1.

In Maaradji et al.'s work one type of social networks is developed. It is referred to as recommendation (due to lack of space limitations of recommendation systems like cold start are not discussed). In Maamar et al.'s work three types of social networks are developed. They are referred to as collaboration, competition, and substitution. Fig. 2 illustrates these social networks types. We recall that 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 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. e.g., node deletion that affects outgoing/incoming edges.

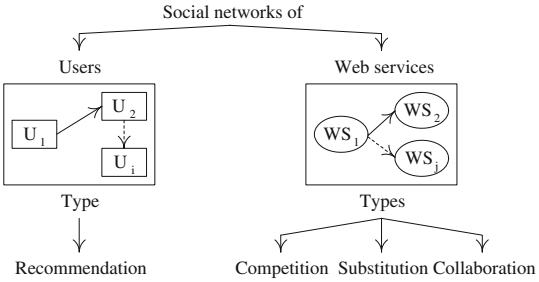


Fig. 2. Social networks of users *versus* social networks of Web services

Social Network of Users. As stated before, recommendation is the sole social network of users that is built to support users develop composite Web services. This network suggests Web services according to the current status of the composition process. The Recommendation Confidence ($\mathcal{RC}$) as discussed in [9] is defined in Equation 1.

$$\mathcal{RC}(ws_k, ws_l) = \sum_{j=1}^{n} \mathcal{NC}_{v_j}(ws_k, ws_l) \times \mathcal{F}it(v_j, ws_l) \times \mathcal{SP}(v_i, v_j) \qquad (1)$$

where:

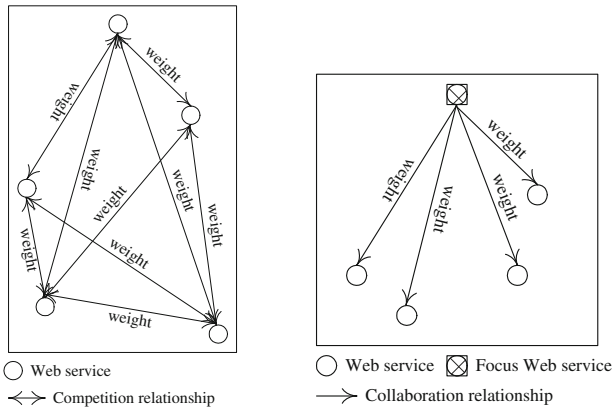
- $\mathcal{NC}_{vj}(ws_k, ws_l)$ represents how many times user v_j used Web service ws_l following the use of Web service ws_k in compositions.
- $\mathcal{F}it(v_j, s_l)$ quantifies the expertise of user v_j in using Web service ws_l.
- $\mathcal{SP}(v_i, v_j)$ defines v_i's social proximity to v_j in the recommendation network.

Social Networks of Web Services. As stated before, collaboration, competition, and substitution are the social networks of Web services that are built to support the development of composite Web services. They are established based on the functionalities of Web services like `checkWeatherForecast` and `bookAirTicket`. Different techniques assess either the similarity or the complementarity of Web services' functionalities. This is outside this paper's scope and interested readers are referred to [3,12]. In the following, the three types of social networks of Web services are explained:

Competition Social Network. Fig. 3 (a) illustrates a competition social network of Web services. Since this network involves Web services that are similarly functional, they are all in competition against each other and hence, all connected to each other through bidirectional edges.

To evaluate the weight of a competition edge, which we refer to as *Competition Level* ($\mathcal{L}_{\mathcal{C}omp}$, Equation 2) between two Web services ws_i and ws_j, we use the Functionality Similarity Level ($\mathcal{L}_{\mathcal{FS}}$) to compare their respective functionalities and the No-Functionality Similarity Level ($\mathcal{L}_{\mathcal{NFS}}$) to compare their respective non-functional properties (*a.k.a* QoS). We assume that the non-functional properties of Web services are defined with the same taxonomy.

$$\mathcal{L}_{\mathcal{C}omp}(ws_i, ws_j) = \mathcal{L}_{\mathcal{FS}}(ws_i, ws_j) \times (1 - \mathcal{L}_{\mathcal{NFS}}(ws_i, ws_j)) \tag{2}$$



(a) Competition **(b) Collaboration**

Fig. 3. Illustration of social networks of Web services

where:

- $\mathcal{L}_{\mathcal{FS}}(ws_i, ws_j)$ corresponds to the similarity level between the respective functionalities of ws_i and ws_j.
- $\mathcal{L}_{\mathcal{NFS}}(ws_i, ws_j) = \omega_1 \times (|\mathcal{P}(ws_{i,1}) - \mathcal{P}(ws_{j,1})|) + \cdots + \omega_n \times (|\mathcal{P}(ws_{i,n}) - \mathcal{P}(ws_{j,n})|)$ with $\mathcal{P}(ws_{i,k})$ is the value of the k^{th} non-functional property of the i^{th} Web service (assumed to be between 0 and 1), ω_k is a weighting factor representing the importance of a non-functional property, and $\sum_{k=1}^{n} \omega_k = 1$.

As per Equation 2 the more the competition level is close to one, the closer ws_i is to ws_j. As a result, ws_i threatens the competitiveness capacity of ws_j. We recall that only one Web service is selected at a time to complete a task in a composition.

Substitution Social Network. Fig. 3 (a) also illustrates a substitution social network of Web services after changing the edges' name from competition to substitution. Since all the Web services in a substitution social network offer the same functionality, any peer is a potential candidate to replace a failing Web service. To evaluate the weight of a competition edge, which we refer to as *Substitution Level* ($\mathcal{L}_{\mathcal{S}ub}$, Equation 3) between ws_i and ws_j, we use like previously the Functionality Similarity Level ($\mathcal{L}_{\mathcal{FS}}$) and the No-Functionality Similarity Level ($\mathcal{L}_{\mathcal{NFS}}$), in addition to the Reliability Level ($\mathcal{L}_{\mathcal{R}}$) that shows how successful ws_i is when it replaces ws_j.

$$\mathcal{L}_{\mathcal{S}ub}(ws_i, ws_j) = \mathcal{L}_{\mathcal{FS}}(ws_i, ws_j) \times \mathcal{L}_{\mathcal{R}}(ws_i, ws_j) \times (1 - \mathcal{L}_{\mathcal{NFS}}(ws_i, ws_j)) \quad (3)$$

where:

- $\mathcal{L}_{\mathcal{FS}}(ws_i, ws_j)$ and $\mathcal{L}_{\mathcal{NFS}}(ws_i, ws_j)$ are defined in Equation 2.
- $\mathcal{L}_{\mathcal{R}}(ws_i, ws_j) = \frac{\mathcal{SR}(ws_i, ws_j)}{\mathcal{TR}(ws_i, ws_j)}$, with $\mathcal{SR}(ws_i, ws_j)$ as the total number of successful replacements that ws_i made for ws_j (i.e., no failure) and $\mathcal{TR}(ws_i, ws_j)$ as the total number of requests that ws_i received to replace ws_j.

Collaboration Social Network. Fig. 3 (b) illustrates a simple collaboration social network of Web services. It is built when at least one composition of Web services is complete. For navigation purposes, an entry node is required and represented differently from the rest of nodes. We refer to this entry node as "focus" Web service. All the edges that come out of the "focus" Web service are unidirectional pointing towards other peers. To evaluate the weight of a collaboration edge, which we refer to as *Collaboration Level* ($\mathcal{L}_{\mathcal{C}ol}$, Equation 4) between ws_i ("focus") and ws_j, we track the number of times that both Web services participated in joint compositions with emphasis on the total number of compositions that ws_i took part in.

$$\mathcal{L}_{\mathcal{C}ol}(ws_i, ws_j) = \frac{\mathcal{JC}(ws_i, ws_j)}{\mathcal{TP}(ws_i)} \quad (4)$$

where $\mathcal{JC}(ws_i, ws_j)$ is the total number of participations of ws_i and ws_j in joint compositions and $\mathcal{TP}(ws_i)$ is the total number of participations of ws_i in compositions.

3.3 Social Composer

In Maaradji et al.'s work [9], a social composer advises the user on the next actions to take with respect to the progress of the composition under construction. This progress depends on the Web services selected recently and appended into this composition. The social composer's examples of advice concern the next suitable Web services to select and the necessary data mappings between the pre- and post-Web services. The social composer uses this user's social network to build recommendation strategies for Web services (Fig. 2). More details on advice and data mapping are given in [9].

Relying only on users' recommendations may not be enough to achieve high-quality compositions. Indeed these recommendations do not capture some Web services' characteristics, e.g., (i) refusal (or conditional refusal) to take part in additional compositions as discussed in [6] and (ii) favoritism for some peers over others as discussed in [5]. For each characteristic a social network of Web services provides specific solutions:

Refusal: Since Web services have to maintain a certain QoS level as per the different Service Level Agreements (SLAs) they commit to, it happens that Web services either decline participation requests in forthcoming compositions or delay their participations until further notice. These requests originate from the social composer. In either case a competition social network can be very useful for the social composer. It helps the social composer identify the peers that can be interested in accepting these participation requests. Without a competition social network, a process for discovering Web services has to be launched from scratch, which is time consuming [7]. A competition social network offers direct access to a pool of candidate peers that are competitors to those that decline participation requests. In the businesswoman example, `museumVisitWS` can decline the social composer's request, which makes the social composer look for a competitor such as `museumTourWS` using the competition social network of `museumVisitWS`.

Favoritism: Since Web services are developed separately, semantic and policy compatibility conflicts will for sure arise [14,17]. To minimize the efforts put into addressing these conflicts, a collaboration social network can be useful for the social composer by identifying the peers that participated with the existing Web services in common compositions. In the businesswoman example, `museumVisitWS` prefers working with `taxiBookingWS` due to successful previous experiences.

3.4 Social Executor

Although the social executor does not advise users like the social composer does, its role is to assess the impact of the social composer's advice on the progress of compositions, so that the relevant social networks of users and of Web services are updated based on this impact. The various advice are about proposing Web services (i) that can take part in compositions in replacement of those that decline participation invitations in these compositions, and (ii) that can achieve a better compatibility level to compositions by minimizing and avoiding semantic and policy conflicts. In Section 3.3, these two cases are referred to as refusal and favoritism, respectively.

Refusal: On top of the details on the competitiveness level ($\mathcal{L}_{\mathcal{C}omp}(ws_i, ws_j)$) that a competition social network carries, this network carries additional details on how Web services responded to the invitations of taking part in compositions since these Web services were not selected initially, i.e., they were less competitive. Therefore, the competitiveness level should be revised ($\mathcal{L}_{\mathcal{RC}omp}(ws_i, ws_j)$) as per the actions that the social executor performs during a composition progress.

Equation 5 is the revised competitiveness level where ws_i accepts a composition invitation that ws_j declines earlier, α_k is a weighting factor ($\sum \alpha_k = 1$), $\mathcal{L}_{\mathcal{C}omp}(ws_i, ws_j)$ is the initial competitiveness level as per Equation 2, $\mathcal{AI}(ws_i, ws_j)$ is the total number of accepted invitations that ws_i received from the social composer following the rejection of ws_j, $\mathcal{TI}(ws_i, ws_j)$ is the total number of invitations that ws_j declined and then, were sent to ws_i, and $\mathcal{P}er(ws_i)$ is the performance of ws_i at run-time using a scale of high, average, and poor. Poor performance means that ws_i "disappointed" the social composer for reasons such as being malicious or inability of delivering the non-functional properties that it posts.

$$\mathcal{L}_{\mathcal{RC}omp}(ws_i, ws_j) = \alpha_1 \mathcal{L}_{\mathcal{C}omp}(ws_i, ws_j) + \alpha_2 \frac{\mathcal{AI}(ws_i, ws_j)}{\mathcal{TI}(ws_i, ws_j)} + \alpha_3 \mathcal{P}er(ws_i) \tag{5}$$

Favoritism: Like in the refusal case, the social composer needs to update both the recommendation social network of users and the collaboration social network of Web services. We focus on the latter type of social network hereafter, which requires reviewing the collaboration level of Equation 4 into $\mathcal{L}_{\mathcal{RC}ol}(ws_i, ws_j)$.

Equation 6 is the revised collaboration level where ws_i takes part in a composition as per the collaboration it has with ws_j, α_k is a weighting factor ($\sum \alpha_k = 1$), $\mathcal{L}_{\mathcal{C}ol}(ws_i, ws_j)$ is the initial collaboration level as per Equation 4, $\mathcal{AC}(ws_i, ws_j)$ is the total number of accepted collaboration requests that ws_i received from the social composer because of the collaboration links it has with ws_j, $\mathcal{TC}(ws_i, ws_j)$ is the total number of collaboration requests that were sent to ws_i because of ws_j, and $\mathcal{P}er(ws_i)$ is the performance of ws_i.

$$\mathcal{L}_{\mathcal{RC}ol}(ws_i, ws_j) = \alpha_1 \mathcal{L}_{\mathcal{C}ol}(ws_i, ws_j) + \alpha_2 \frac{\mathcal{AC}(ws_i, ws_j)}{\mathcal{TC}(ws_i)} + \alpha_3 \mathcal{P}er_{ws_i} \tag{6}$$

3.5 Social Monitor

The social monitor comes into play when a Web service fails so a substitute needs to be found. To this end the social monitor uses the substitution social network of the failing Web service to identify the most appropriate substitute(s) (comparing $\mathcal{L}_{\mathcal{C}omp}(ws_i, ws_j)$ to a threshold). Two cases arise: (i) the substitute Web service accepts the request of the social monitor and hence, joins the composition that is put on-hold and now needs to be resumed; and (ii) the substitute Web service rejects the request of the social monitor and hence, another substitute needs to be identified. In either case, the substitution level is updated. Furthermore the collaboration level is updated for the first case.

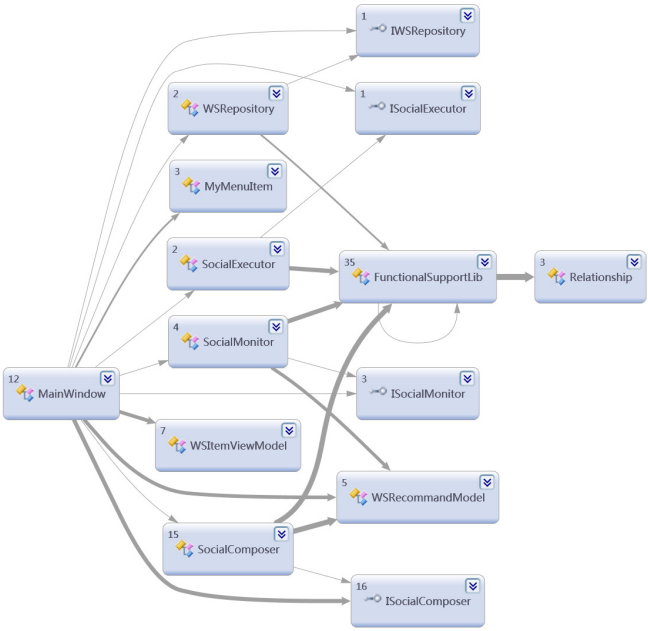


Fig. 4. Main classes for the system prototype

Equation 7 is the revised substitution level where ws_i replaces ws_j, α_k is a weighting factor ($\sum \alpha_k = 1$), $\mathcal{L}_{\mathcal{S}ub}(ws_i, ws_j)$ is the initial substitution level as per Equation 3, $\mathcal{AS}(ws_i, ws_j)$ is the total number of accepted substitution requests that ws_i received from the social monitor because of the substitution links it has with ws_j, $\mathcal{TS}(ws_i, ws_j)$ is the total number of substitution requests that were sent to ws_i because of the failure of ws_j, and $\mathcal{P}er(ws_i)$ is the performance of ws_i using a scale of high, average, and poor.

$$\mathcal{L}_{\mathcal{RS}ub}(ws_i, ws_j) = \alpha_1 \mathcal{L}_{\mathcal{S}ub}(ws_i, ws_j) + \alpha_2 \frac{\mathcal{AS}(ws_i, ws_j)}{\mathcal{TS}(ws_i)} + \alpha_3 \mathcal{P}er(ws_i) \quad (7)$$

4 Implementation and Evaluation

In this section, we report a prototype implementation and some preliminary experiments demonstrating the feasibility of developing a user-centric social approach to Web services composition, execution, and monitoring.

The prototype is developed using .Net Framework 4, Windows Presentation Foundation (WPF), and Visual Studio 2010 SP1. The main classes forming the prototype and their dependencies are shown in Fig. 4. The `MainWindow` class offers a visual interface of the prototype to users. The `WSRepository` class manages Web services

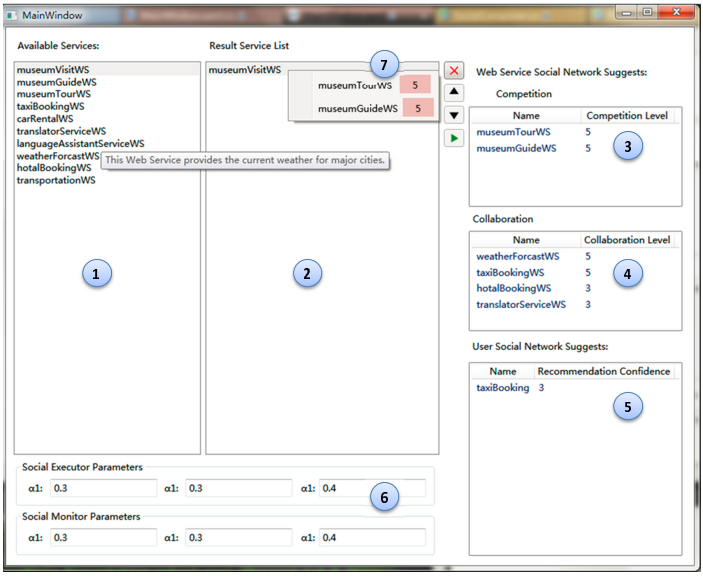


Fig. 5. The system frontend

while the `SocialExecutor`, `SocialMonitor`, and `SocialComposer` achieve the functionalities of the three proposed components in Section 3. Each of these classes is associated with an interface class (e.g., `ISocialExecutor` is the interface class of `SocialExecutor`). The `FunctionalSupportLib` class implements the low level functions of different social networks of users and the ones of Web services. The detailed information of these networks is stored in the `Relationship` class that serves as a data container. Finally, the `MyMenuItem` class specifies how to display customized menu item of the prototype and the `WSItemViewModel` class specifies how to render a Web service item into a listview box.

Fig. 5 shows a snapshot of the prototype system. The left panel (see Part 1) is a container for all available Web services such as `taxiBookingWS`. When double clicking on one of these Web services, the service will be added to the final result list that corresponds to the composite Web service to build (i.e., the middle panel Part 2). By clicking on a Web service in the middle panel (e.g., `museumVisitWS`), the system suggests recommendations through different social networks (see the right panel). For example, `museumTourWS` and `museumGuideWS` are recommended from the competition social network while four others are suggested from the collaboration social network. Each service has a competition or collaboration level value that is calculated by using the formulas developed in Section 3.2. A user can adjust the parameters of the formulas from the bottom panel (see Part 6). At any time, the user can conveniently add Web services from other panels (i.e., Parts 1, 3, 4, 5) to the composition list (i.e., Part 2) by simply double clicking them. If she would like to replace a Web service from Part 2, she can right click on the name of the Web service and a list of similar Web services will pop up (see Part 7), from which a substitute can be chosen.

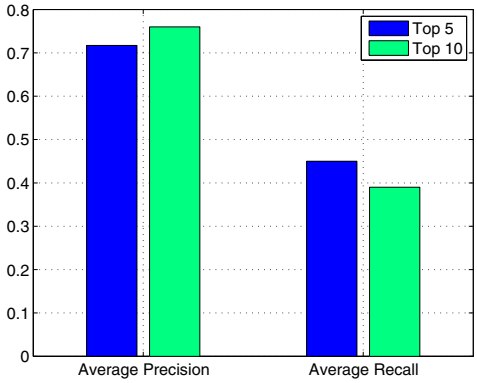


Fig. 6. Precision and recall of the competition social network

Moreover we conducted some preliminary experiments to study the performance of the system. Due to space constraints, we report one of them that studies the recommendation performance of competition social networks in case of refusal during composition. The concepts of *Precision* (P) and *Recall* (R) in information retrieval were used to evaluate the effectiveness of service recommendation. In the experiment, Web services with competition level higher than 7 were considered as valid recommendations. We examined the top N (N=5 and 10) candidates. In Figure 6 we can see that our approach achieves a reasonably good performance (close to 0.8 in precision when N is set to 10). The following was set for the experiment needs. Let p be the number of all Web services that are relevant to the Web service to be substituted, q be the number of relevant Web services recommended, and r be the total number of Web services recommended, then $R = q/p \times 100$ and $P = q/r \times 100$. A high precision value means that there are few false alarms while a high recall value means that there are few false dismissals.

5 Conclusions and Future Work

This paper has discussed the intertwine of social networks of users and social networks of Web services. The former contain details that help users select the necessary Web services when building composite Web services while the latter contain details that permit extending these composite Web services with new Web services or maintaining the operation continuity of these composite Web services in case of failure. This social networks intertwine is taken care by three social components referred to as composer, executor, and monitor. Each performs different operations. For instance the social composer suggests which Web services to check in case some peers decline to participate in composite Web services, the social executor assesses the impact of the social composer's advice on composite Web services execution progress, and last but not least the social monitor feeds the social executor with details so that this latter updates the social networks of Web services for the benefit of the social composer. A prototype has been developed to demonstrate the feasibility of developing a user-centric social approach to Web services composition, execution, and monitoring. Different future research works

are identified including adding a social adaptor to the general approach and assessing the overhead of taking into account the different social networks on the progress and performance of composite Web services as well as the quality of the composite Web services that are developed.

References

1. Al-Sharawneh, J., Williams, M.A.: A Social Network Approach in Semantic Web Services Selection using Follow the Leader Behavior. In: Proc. of the 13th Enterprise Distributed Object Computing Conference Workshops (EDOCW 2009), Auckland, New Zealand (2009)
2. Bengtsson, M., Kock, S.: Coopetition in Business Networks to Cooperate and Compete Simultaneously. Industrial Marketing Management 29(5) (2000)
3. Di Martino, B.: Semantic Web Services Discovery based on Structural Ontology Matching. International Journal of Web and Grid Services 5(1) (2009)
4. Maamar, Z., Faci, N., Krug Wives, L., Yahyaoui, H., Hacid, H.: Towards a Method for Engineering Social Web Services. In: Ralyté, J., Mirbel, I., Deneckère, R. (eds.) ME 2011. IFIP AICT, vol. 351, pp. 153–167. Springer, Heidelberg (2011)
5. Maamar, Z., Hacid, H., Hunhs, M.N.: Why Web Services Need Social Networks. IEEE Internet Computing 15(2) (March/April 2011)
6. Maamar, Z., Kouadri Mostéfaoui, S., Yahyaoui, H.: Towards an Agent-based and Context-oriented Approach for Web Services Composition. IEEE Transactions on Knowledge and Data Engineering 17(5) (May 2005)
7. Maamar, Z., Wives, L.K., Badr, Y., Elnaffar, S., Boukadi, K., Faci, N.: LinkedWS: A Novel Web Services Discovery Model Based on the Metaphor of "Social Networks". Simulation Modelling Practice and Theory 19(10) (2011)
8. Maamar et al., Z.: Using Social Networks to Web Services Discovery. IEEE Internet Computing 15(4) (July/August 2011)
9. Maaradji, A., Hacid, H., Daigremont, J., Crespi, N.: Towards a Social Network Based Approach for Services Composition. In: Proceedings of the 2010 IEEE International Conference on Communications (ICC 2010) (2010)
10. Maaradji, A., Hacid, H., Skraba, R., Vakali, A.: Social Web Mashups Full Completion via Frequent Sequence Mining. In: Proceedings of the IEEE 7th World Congress on Services (SERVICES 2011), Washington DC, USA (2011)
11. Menascé, D.A.: QoS Issues in Web Services. IEEE Internet Computing 6(6) (November/December 2002)
12. Min, L., Weiming, S., Qi, H., Junwei, Y.: A Weighted Ontology-based Semantic Similarity Algorithm for Web Services. Expert Systems with Applications 36(10) (December 2009)
13. Nam Ko, M., Cheek, G.P., Shehab, M., Sandhu, R.: Social-Networks Connect Services. IEEE Computer 43(8) (August 2010)
14. Paolucci, M., Kawamura, T., Payne, T.R., Sycara, K.: Semantic Matching of Web Services Capabilities. In: Horrocks, I., Hendler, J. (eds.) ISWC 2002. LNCS, vol. 2342, pp. 333–347. Springer, Heidelberg (2002)
15. Papazoglou, M., Traverso, P., Dustdar, S., Leymann, F.: Service-Oriented Computing: State of the Art and Research Challenges. IEEE Computer 40(11), 38–45 (2007)
16. Pedrinaci, C., Domingue, J.: Toward the Next Wave of Services: Linked Services for the Web Data. Journal of Universal Computer Science 16(13) (2010)
17. Sheng, Q.Z., Yu, J., Maamar, Z., Jiang, W., Li, X.: Compatibility Checking of Heterogeneous Web Service Policies Using VDM++. In: Proc. of the 2009 IEEE Congress on Services, Part I (SERVICES I 2009) (2009)

18. Tan, W., Zhang, J., Foster, I.: Network Analysis of Scientific Workflows: A Gateway to Reuse. IEEE Computer 43(9) (September 2010)
19. Wu, Q., Iyengar, A., Subramanian, R., Rouvellou, I., Silva-Lepe, I., Mikalsen, T.: Combining Quality of Service and Social Information for Ranking Services. In: Baresi, L., Chi, C.-H., Suzuki, J. (eds.) ICSOC-ServiceWave 2009. LNCS, vol. 5900, pp. 561–575. Springer, Heidelberg (2009)
20. Xie, X., Du, B., Zhang, Z.: Semantic Service Composition based on Social Network. In: Proc. of the 17th Intl. World Wide Web Conf. (WWW 2008), Beijing, China (2008)
21. Yu, J., Sheng, Q.Z., Han, J., Wu, Y.: A Semantically Enhanced Service Repository for User-Centric Service Discovery and Management. Data & Knowledge Engineering 72(1) (2012)