SOAF – Design and Implementation of a Service-Enriched Social Network[⋆]

Martin Treiber, Hong-Linh Truong, and Schahram Dustdar

Distributed Systems Group, Institute of Information Systems
Vienna University of Technology
{treiber,truong,dustdar}@infosys.tuwien.ac.at

Abstract. In this paper, we propose the integration of services into social networks (SOAF - Service of a Friend) to leverage the creation of the Internet of Services vision. We show how to integrate services and humans into a common network structure and discuss design and implementation issues. In particular, we discuss the required extensions to existing social network vocabulary with regard to services. We illustrate a scenario where this network structures can be applied in the context of service discovery and highlight the benefit of a service-enriched social network structure.

1 Introduction

The Internet of services [1] vision focuses on the extension of the existing Internet with regard to services. In a future Internet of services, information is not static any more, but dynamically provided by all kind of software services. This development was driven by the so-called Web 2.0 phenomena, which included the broad adoption of social networks like facebook [1], xing [2] or twitter[3]. Indeed, as Kleinberg observed in his work [2], social and technical networks converge. In these networks, user generated content, like folksonomies [3], provides a vast source of information that is able to classify arbitrary content (e.g., del.icio.us [4]). In this area, the Friend of a Friend project (FOAF) [4] aims at providing information about relationships between humans in social network structures. FOAF describes relationship structures with RDF [5], thus defining the technical foundation to access information of social networks in a machine readable form.

Viewed from a business perspective, these developments have a profound impact on the way businesses are conducted. In his Wired article, Howe shows how the idea of crowdsourcing [6] can be applied to businesses. With regard to (Web) services, which already provided by companies, and the integration of humans into common networks, companies can benefit from these emerging network structures. However, with existing service-oriented infrastructures, this endeavor proves to be difficult to achieve. In

⋆ The research leading to these results has received funding from the European Community's Seventh Framework Programme FP7/2007-2013 under grant agreement 215483 (S-Cube).

1 `http://www.facebook.com`

2 `http://www.xing.com`

3 `http://www.twitter.com`

4 `http://delicious.com`

fact, SOA (service-oriented architecture) focuses on stable processes that are defined and executed in workflow systems. The gap between SOA and Web 2.0 is widened by the emerging end user driven creation of applications (mashups [7] and situational applications [8,9]) which are playing an important part on the Internet now and become increasingly important for businesses.

One of the main reasons is that there is little support to integrate humans and services into networks to benefit from social connections within such network structures. There exist approaches that support the integration of human activities [10,11] into business processes. These approaches assume that there is already a workflow and that there is repository that can be used to select the required services for a given workflow. The associated service discovery process is well studied in literature [12]. However, with the failure of centralized registries [13] and no Web service standard for the discovery of Web services, the discovery process is fragmented and cumbersome. This leads to a situation in which Web service related information is distributed among several isolated company registries, if this is the case at all. Especially smaller companies hesitate to use registries, because of the overhead involved in maintaining dedicated registries. In such cases, Web services are often published simply by mailing customers the necessary information about the endpoint of a Web service or maintaining simple catalogues with unstructured information of available Web services on company owned web pages.

This practice hinders the creation of Web service marketplaces [14] where one can discover Web services and learn from the experience of others by using a particular Web service. When investigating the process of Web service discovery, one finds that the human factor is dominant in (semi-) automated approaches [15]. Furthermore, structured meta information in form of ontologies [16] suffers from the same limitations concerning availability as centralized registries. Even with available semantic information, the process of discovering Web services requires human activities, since different semantic service descriptions can be provided by different ontologies. These ontologies require mappings which cannot be fully automated due to ambiguities or even contradictions within their content [17].

In the context of Web service discovery, we can learn lessons from humans and how they look for solutions of problems. Humans exploit local information and use links to other persons to ask for pointers or for information when needed. In short, humans ask their friends whether they had a similar problem and how the problem was solved. In our work, we aim to make use of human relations together with service information. We link software services and humans in a common network structure. We refer to this approach as *Service of a Friend* (SOAF) and follow the spirit of FOAF. We believe that the integration of humans and services into networks fosters the creation of Web service ecosystems [18]. Our approach bears several challenges that we are going to address in this paper. First of all, we need a representation of the links between services and humans. Secondly, dynamic changes must be represented in our network, since there are relations that exist only over a certain time (e.g., projects may require collaboration for several months). And finally, we need to consider that past relations provide useful information for potential future use (e.g., a service that was useful for certain tasks in the past may be again useful for new tasks of different users).

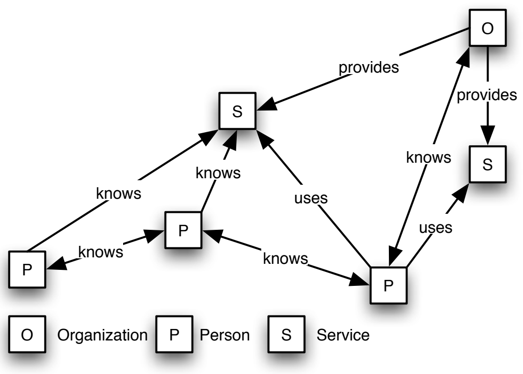

Fig. 1. Overview of SOAF network structure

The rest of the paper is organized as follows. We discuss our approach in Section 2. We provide an analysis and discussion of our findings in Section 3. Afterwards, we introduce our prototype architecture in Section 4. We conclude our paper with related work in Section 5 and an outlook for future research directions in Section 6.

2 Linking Web Services

Due to distributed nature of services and the lack of centralized repositories to search for services, we require meta information that provides information about the connections (links) between services in service networks (see Figure 1). These network structures originate for instance from organizational structures of companies or social networks which model social connections between humans. Thus, links and their associated information are very critical for the traversal of networks efficiently and to facilitate the discovery of distributed services. Therefore, we include meta information into links to make the traversal more efficient. Furthermore, as the linkage between elements of networks is constantly changing, we consider dynamic aspects of the relations between services, organizations and humans as well. These are not static and may change over time. For instance, a person might move from one organization to another or the service provision might depend on the duration of a project (e.g., event notification services). Our approach takes these considerations into account and we discuss our concept in detail in the following sub-sections.

2.1 Extending FOAF

The integration of services and humans in a common information network requires the integration of existing social network structures and service related information. Our idea is to augment FOAF network structures with service related information and to link services and humans in the same network. In particular, we extend the relation mechanisms of FOAF to model relations between services and persons. In SOAF [5] we

[5] `http://www.infosys.tuwien.ac.at/staff/treiber/soaf/index.rdf`

extend the FOAF concepts with a (i) *Service* concept to represent services, a (ii) *uses* relation which denotes the use of a service by a person or other services (iii) and a *provides* relation that specifies the relation between service provider and service. With these extensions, we can establish relations between persons, services and organizations/groups. We summarize the relations in Table 1.

Table 1. Relations between SOAF entities

Relation	Description
Service uses service	Denotes direct service invocation by other services. For instance, in service compositions, a service might call another service directly
Service knows service	Denotes that two services are related within a certain context (e.g., workflows, compositions or mashups) without any direct invocation of each other
Person uses service	Denotes the service use of a person
Person knows service	Denotes mutual knowledge of a service and a person without usage
Person provides service	Denotes the service provision by a person, e.g., a human provided service
Organization provides service	Defines the relation between organizations and their provided services

2.2 Dynamic SOAF

As discussed before, we model three basic relations between entities in a SOAF network (i) knows, (ii) uses and (iii) provides. The latter two imply automatically knows, since it is required to know a service before it can be offered or consumed. *Knows*, *uses* and *provides* are pairwise related through a simple subset relation: *uses* is a subset of *knows*, since it is required to know a service before a service can be used. Besides, persons/organizations might know more services than they actually use. The *provides* relation is also a subset of the knows relation, since a provider knows obviously the services that are provided and knows/uses additional services.

Viewed from a time based perspective, elements of the *knows/uses/provides* sets are subject to changes. For instance, a service might move from the *knows* set to the *uses* set and vice versa. Consequently, we allow to have multiple links to a single service from a person at any point in time. For instance, as soon as a service is used by a person, a *uses* relation is created. If the service is not used anymore (e.g., the service was used for registration purposes or the access has been revoked due to company changes, etc.) the *uses* relation is not valid any more and its internal state and timestamp are set accordingly. Thus the service moves to the *knows* set.

Notice that the *knows* relation is static: once a person knows another person/service, the relation remains - it is not removed anymore. However, with services we have to pay attention to the fact that a service does simply not exist any more. In these cases, the *knows* relation points to an inactive services that have been used in the past.

An aspect that needs explicitly to be considered is the type of service usage. We identified several types of service usage that we include in our model. We use this kind of information to generate accurate historical information. In particular we consider the usage frequency of a service and classify the usage as summarized in Table 2.

Table 2. Service usage in SOAF

Usage	Description
Once	The service is only used once and then never again during the lifetime of the service (e.g.,a registration/unregistration service is used to subscribe to a mailinglist, a polling service might exist only before a certain event takes place, etc.).
Continuously with pre-defined time to live	The service is used for a certain activity during a pre defined time and is removed afterwards (e.g., a service that provides state information about persons in a project).
Continuously	The service is used continuously without limitations concerning the time of use and frequency.

Complementary to the use of services is their provision. Service provision changes also over the time, but is generally less dynamic than the *uses* relation. In particular we consider three distinct service provision scenarios that are supported by our model (see Table 3).

Table 3. Service provision in SOAF

Usage	Description
Continuously with pre-defined time to live	The service is provided for a certain activity during a pre defined time and is removed afterwards (e.g., a service that provides state information about persons in a project).
Continuously	The service is provided continuously without limitations concerning the time of use and frequency. This includes the case when a service is used only once for registration purposes, but nevertheless is required by different customers to register and thus must be available continuously.
Deprecated	The service is still available but not actively maintained,

Of central importance for the representation of the network dynamics is the connection between entities in the SOAF network. We include additional meta information in the linkage of SOAF entities that is important for management purposes (e.g., creation and deprecation) (see Table 4).

Table 4. SOAF Connection attributes

Attribute	Description
Creation	Date, on which the connection between the entities was established
Removal	Date on which the connection was removed
Active	Flag that indicates if a connection is currently active
Type	Defines the type of connection, either *uses*, *provides* or *knows*

2.3 Managing SOAF Service Networks

The management of dynamic aspects of distributed networks is complex task. The first challenge is to identify a resource in a network in a unique manner. In our approach, we follow the concept of ”inverse functional properties” from OWL [19]. We use a functional property that defines the URI of a person, an organization or a service. Services include a functional property that points to the endpoint of the service as well.

Since we do not intend to define a centralized authority that manages available information, we have to rely on all network members to manage their links and to keep the links updated. Still, there is no guarantee that this process works without disruption, since this process relies partially on the intervention of humans. However, since links between entities in the SOAF network imply a certain degree reciprocal agreement (*knows* relation, *uses* relation) we support this by including Atom feed based [20] notification mechanisms. This is an extension to our previous work on service evolution management (SEMF) [21] that is able to manage distributed information of services that change during their life-cycle. Like SEMF, SOAF supports a set of events that can be subscribed to and that can be accessed as Atom feeds (see Table 5).

Table 5. SOAF events

Event	Description
Registration	This event describes the creation of a new SOAF entity in the SOAF network. This event is generated upon the creation of a service, a person or an organization
Change	This event describes changes of SOAF entities
Removal	This event is generated upon the removal of a SOAF entity
Connection	This events is generated if a new connection between two SOAF entities is established

Service Publication, Service Removal. Service publication in SOAF is done locally. A service provider updates its SOAF description with new services that are offered. Since we do not have a central entity that is used for service registration, we do not require service providers to actively contact a registry and provide information. Other SOAF network members that are registered for service publication events receive corresponding notifications, i.e., a registration event that contains service related information. If a network member is interested in this newly registered service, the network member contacts the service after having received the registration event and asks for the service profile. The protocol is shown in Figure 2.

The removal of existing service is closely related to the publication process. To remove a service the provider deletes the service information from its local SOAF description. The propagation of the update follows the same pattern as the publication with regard to the propagation of changes. Upon service removal, the provider obtains a list of all service users. Then, the provider checks its subscribed SOAF network members and informs all service users and the subscribed network members about the service removal with a removal event (see Figure 3).

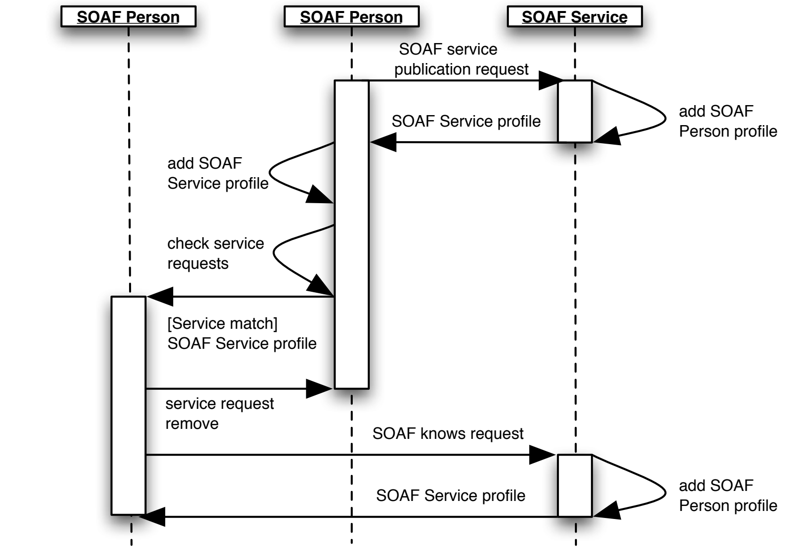

Fig. 2. SOAF publication information protocol

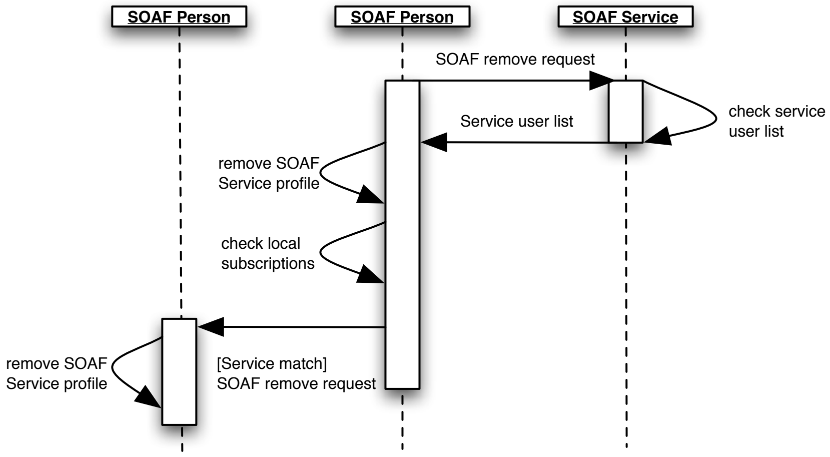
SOAF Person
SOAF Person
SOAF Service
SOAF remove request
check service user list
Service user list
remove SOAF Service profile
check local subscriptions
remove SOAF Service profile
[Service match] SOAF remove request

Fig. 3. SOAF removal protocol

It it worth noticing that from a conceptual point of view, SOAF does not limit this approach to humans. Since we envision services as part of the network, and thus providing well formed information, we can extend the notification to services as well.

2.4 Extending the FOAF Datamodel

In this section, we discuss the extensions of the FOAF data model with regard to SOAF concepts. Notice that the mapping is not limited to FOAF in particular, other representations of SOAF concepts are also possible. An alternative could be the use of XML structures that are linked with XLink constructs [22]. However, since FOAF has gained considerable adoption [23] [24] we have decided to integrate our prototype data model into the FOAF data model. SOAF requires new concepts to be added to the main FOAF data model with regard to the needs of services. We include a (i) *Service* class to represent services that inherits from Agent, a (ii) *uses* relation which is similar to the *knows* relation, but provides additional information, (iii) a *provides* relation that defines the connection between service providers (which may be organizations, persons, teams, virtual teams), and (iv) a dedicated *Connection* class (which also inherits from Agent) that encapsulates the connection between services, persons and organizations (see Figure 4).

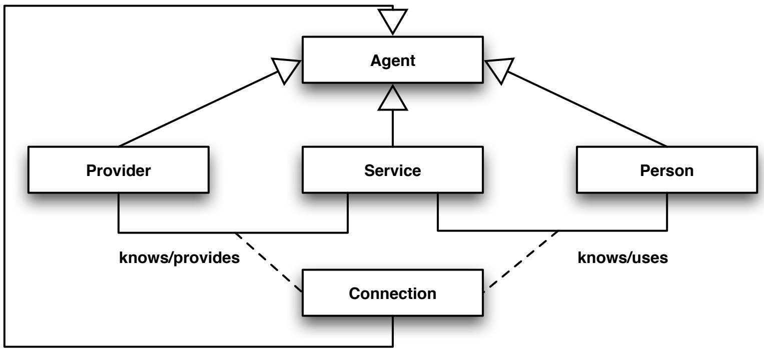

Fig. 4. An implementation moel of the SOAF network structure

SOAF Service Class. We model service related information in the SOAF service class. In our prototype data model, we provide a basic set of information that defines the capabilities service. The SOAF service class offers information about the endpoint of the service, the interface description, version information, etc. (see Listing 1.1).

```
<soaf:Service>
  <foaf:name>SOAFer</foaf:name>
  <soaf:endpoint>...</soaf:endpoint>
  <soaf:description>SOAF Service Profiles Generator</soaf:description>
  <soaf:interface rdf:resource="..."/>
  <soaf:active>true</soaf:active>
  <soaf:version>1.0</soaf:version>
</soaf:Service>
```

Listing 1.1. SOAF service class example snippet

SOAF Connection Class. The introduction of the connection class addresses the major shortcoming of FOAF with regard to connections between persons and services. In our data model, we need to attach additional attributes to a connection like creation date, state of the connection, etc. FOAF uses the *knows* relation to connect persons and this relation does not support additional attributes to further refine the type of connection.

Thus, we modeled connection class as a subclass of the *agent* class. This allows us to seamlessly integrate SOAF connections using the *knows* relation as bridge to FOAF. The connection class acts as a container for the connection between persons and services (see Listing 1.2).

```
<foaf:knows>
  <soaf:Connection>
    <soaf:established>January 23rd 2009</soaf:established>
    <soaf:active>true</soaf:active>
    <soaf:connectiontype>Continuous</soaf:connectiontype>
    <soaf:uses>
      <soaf:Service>
         <foaf:name>SOAFer</foaf:name>
          ...
      </soaf:Service>
    </soaf:uses>
  </soaf:Connection>
  <soaf:Connection>
    <soaf:established>December 1st 2008</soaf:established>
    <soaf:discontinued>December 21st 2008</soaf:discontinued>
    <soaf:active>false</soaf:active>
    <soaf:connectiontype>Continuous</soaf:connectiontype>
    <soaf:uses>
      <soaf:Service>
         <foaf:name>SOAFReporter</foaf:name>
          ...
      </soaf:Service>
    </soaf:uses>
  </soaf:Connection>
</foaf:knows>
```

Listing 1.2. SOAF connection class example snippet

SOAF Uses Relation. The *uses* relation is encapsulated in the SOAF connection class and denotes the service usage of persons, providers and services (see Listing 1.3 for an example of the *uses* relation).

```
<foaf:knows>
  <soaf:Connection>
    <soaf:established>January 23rd 2009</soaf:established>
    <soaf:active>true</soaf:active>
    <soaf:connectiontype>Continuous</soaf:connectiontype>
    <soaf:uses>
      <soaf:Service>...</soaf:Service>
    </soaf:uses>
  </soaf:Connection>
</foaf:knows>
```

Listing 1.3. SOAF uses example snippet

SOAF Provides Relation. Like the *uses* relation, the *provides* relation is encapsulated in the connection class. The *provides* relation describes connections between providers and their services where every connection models the provision of a service (see Listing 1.4).

```
<foaf:knows>
  <soaf:Connection>
    <soaf:established>January 23rd 2009</soaf:established>
    <soaf:active>true</soaf:active>
    <soaf:deprecationdate>July 23rd 2009</soaf:deprecationdate >
    <soaf:connectiontype>Continuous</soaf:connectiontype>
    <soaf:provides>
      <soaf:Service>... </soaf:Service>
    </soaf:provides>
  </soaf:Connection>
</foaf:knows>
```

Listing 1.4. SOAF provides example snippet

3 Discussion

One of the major benefits of the SOAF network is that we are able to create a dynamic ecosystem of services from a bottom up approach. In particular, since we integrate humans and services alike, we can track relations between different stakeholders of Web services [25]. For instance, a service developer might integrate different services into a new service by wiring the respective service invocations in the code of the service. By storing such information into SOAF networks, we provide information about service dependencies and input for creating dependency graphs of services.

Another important aspect is to consider historical information in SOAF which are particular interest for service mashups. These are created for a certain purpose, and this kind of information is reflected by connections of different services and persons that used this particular service mashup. Depending on the amount of meta information provided, we provide the ability to search in SOAF networks for examples of mashups that solved particular problems. These examples can be viewed as best practices and thus serve as blueprint for the creation of other mashups.

Related to historical information is the aspect of network evolution. With the data provided by SOAF, we can observe the development of network connections (*uses, knows, provides* relation) and study the general dynamics of the service network. For instance, we can establish the number of services that oined the network during a certain period of time or how many services where removed, etc. Another example is the creation of metrics that define the attractiveness of services for other members of the SOAF network, based on the data SOAF provides.

As ”side-effect” in SOAF, we can observe emerging clusters of well connected services and persons. This allows us to foster communities in a bottom up manner from existing connections between services and persons. In contrast to existing Web service community approaches, we follow the social aspect more closely and do not pre-define the community functionality. We are aware that a social approach brings a certain degree of fuzziness. Furthermore, it is difficult to obtain the overall functionality of communities, since some services might overlap in their functionality. Especially when limited information is available (e.g., WSDL descriptions), a clear description in terms of overall community functionality might not be feasible. However, even with fuzzy information, we are able to define a set of core functions that are used within a community since through the community structure we know which services have the highest connection and usage rates.

SOAF also supports *social based service discovery* which is the translation of human search activities into a service discovery process. To illustrate our approach, consider the following example. Company A needs a service that is able to provide information of public holidays in european countries, for a project meeting planning purposes. Traditionally, an employee of company A would search a public registry or search engine [6] for a service that is able to fulfill this requirements. If no corresponding service can be found, the search is repeated after a while in order to find a service and eventually a service may be found (we assume, that such a service exists in reality and is published during the time the employee searches for it).

When we transform the discovery example from above to a social network oriented approach, person A would ask another person B (colleague from work or friend) if s/he knows a holiday information service. If this is not the case, then person A could ask person B if person B either knows another person that in turn could be asked or if s/he hears from such a service to inform person A about the service. This approach is also known as epidemic protocol [26]. The discovery process we envision in SOAF mimics the process that we described above. First of all, we assume that SOAF provides a link between person A and person B. Furthermore, Person B has connections to services and persons s/he knows and/or uses. By following our example, person A browses all services that person B knows and learns that none of the services known by B is able to provide the required functionality. In this case, person A registers to a feed person B provides in order to get a notification if person B finds a service or if person B is linked with a service of the required functionality. Simultaneously, person A can do the same with other persons in the network and thus distribute the discovery among other network participants by following links.

4 Prototype

We base our prototype on the distributed architecture of previous work [21] and extends it with the required functionality to model SOAF networks. Our prototype uses a XML database [7] to persist SOAF related information, which we organize internally in several different collections (see Table 6). We use XQuery expressions to generate SOAF profiles from the persisted data [8].

Our prototype provides the basic functionality (implemented as REST-based Web services) to manage SOAF data. In order to provide access to events, our prototype generates Atom feeds from SOAF data. We organize the events in three separate feeds as shown in Figure 4. For analytical purposes and to co-relate events, we foresee links between different entries of the feeds.

5 Related Work

From a technical perspective, our approach have similarities with the Web Service Introspection Language [27]. Like WSIL, SOAF also provides a container to store Web

[6] seekda.com, strikeiron.com, xmethods.net

[7] eXist XML Database `http://www.exist-db.org/index.html`

[8] For an example see `http://www.infosys.tuwien.ac.at/staff/treiber/soaf/MartinTreiber.soaf`

Table 6. SOAF collections in the XML Database backend

Collection	Description
Service	Stores service related information (e.g., endpoint, link to interface description, etc.)
Person	Stores all person relevant data (e.g., name, surname, etc.)
Connection	Stores connection information (e.g., knows, uses, provides, etc.)
Organization	Contains information about organizations (e.g., name, address, etc.)

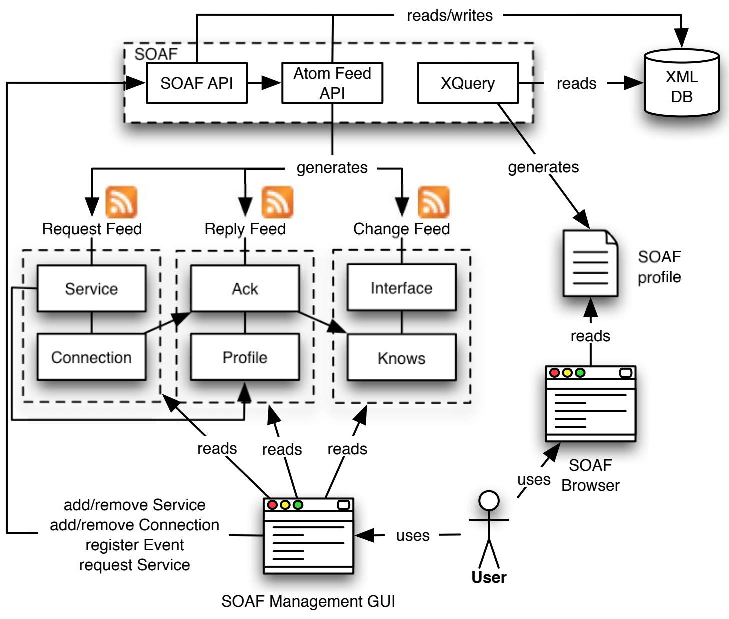

Fig. 5. SOAF prototype framework overview

service related data and supports the linking of services with each other. In contrast to WSIL, SOAF extends the service linkage towards social networks that is not provided by WSIL itself and integrates humans and services into a common network.

Semantic Web service communities as introduced by [28] aim at creating communities of Web services. However, the aforementioned approach focuses on issues like service replaceability and how semantic descriptions of communities can be created. We consider our approach at the other end of the spectrum, since SOAF follows a bottom up approach and doesn't require ontologies to define the available service functionality. Moreover, we explicitly consider humans and services as fundamental part of a network and integrate social structures into of service networks.

The work of Basole and Rouse [29] is related to our work in general. Value Networks [30] are of interest when business aspects are studied, i.e., the value that can be generated by such networks. This is of particular interest when we use our approach to structure available information of humans and services for further analysis with regard to businesses.

Mandelli [31] studies self-organizational aspects that are of importance for our work, since we consider SOAF as environment where we can investigate emergent structures. What distinguishes our approach is the technical focus of our work since we aim to augment existing social networks with service descriptions that we consider this as foundation for the integration of services in a future Internet of Services [32].

Throughout our work, we utilize concepts that originate from connector oriented architectures [33]. In particular, we borrow the concept of connectors to model connections between services and humans in SOAF networks. Furthermore, we also consider dynamic aspects of connections between entities in SOAF networks. With respect to changes, we refer to software evolution which has been studied on software architecture level [34] and evolution languages have been proposed to model software architecture changes. While conceptually similar, our focus lies on the basic support for change mechanisms.

6 Future Work

In this paper, we have presented SOAF (Service of a Friend), which integrates humans and services into a common network structure. We have showed how to model humans and services by extending FOAF and providing a common data model. In future work, we are going to analyze scalability issues in our proposed SOAF network structure. Since we consider humans in the loop we require a simulation model to estimate the human impact in such networks (e.g., during searching). Closely related are human provided services [35] which we are going to investigate in the context of SOAF. In particular we are going to study dynamic aspects like quality of service of human provided services and how to address these issues in SOAF.

Furthermore, we are going to investigate how to generate larger networks from existing data. In order to obtain simulation data, we are going to crawl social networks and to address the important question how to bootstrap SOAF networks from this data. With simulations of larger SOAF networks we are going to study evolutionary aspects of social service networks. Of particular interest is the study of concepts like service fitness in simulations of SOAF networks and the impact analysis of fitness changes in such networks.

References

1. Ruggaber, R.: Internet of services sap research vision. In: WETICE 2007: Proceedings of the 16th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises, Washington, DC, USA, p. 3. IEEE Computer Society, Los Alamitos (2007)
2. Kleinberg, J.: The convergence of social and technological networks. Commun. ACM 51, 66–72 (2008)
3. Voss, J.: Tagging, folksonomy & co - renaissance of manual indexing? CoRR abs/cs/0701072 (2007)
4. Brickley, D., Miller, L.: Foaf vocabulary specification 0.91 (November 2007)
5. W3C: Resource Description Framework (RDF) (2000)

6. Howe, J.: The rise of crowdsourcing (June 2006), http://www.wired.com/wired/archive/14.06/crowds.html
7. Maximilien, E., Wilkinson, H., Desai, N., Tai, S.: A domain-specific language for web apis and services mashups. In: Krämer, B.J., Lin, K.-J., Narasimhan, P. (eds.) ICSOC 2007. LNCS, vol. 4749, pp. 13–26. Springer, Heidelberg (2007)
8. Saphir, J.: Situational applications - cost-effective software solutions for immediate business challenges (2008), http://knol.google.com/k/jonathan-sapir/situational-applications/
9. Shirky, C.: Situated software (2004)
10. Endpoints, A., Systems, A., Systems, B., Corporation, I., Oracle, SAP: Ws-bpel extension for people (bpel4people), version 1.0 (June 2007)
11. Schall, D., Truong, H.L., Dustdar, S.: Unifying human and software services in web-scale collaborations. IEEE Internet Computing 12, 62–68 (2008)
12. Garofalakis, J.D., Panagis, Y., Sakkopoulos, E., Tsakalidis, A.K.: Contemporary web service discovery mechanisms. J. Web Eng. 5(3), 265–290 (2006)
13. Microsoft: Uddi shutdown (2006)
14. van den Heuvel, W.J., Yang, J., Papazoglou, M.: Service representation, discovery, and composition for e-marketplaces. Cooperative Information Systems, 70–284 (2001)
15. Benatallah, B., Hacid, M.S., Leger, A., Rey, C., Toumani, F.: On automating web services discovery. The VLDB Journal 14(1), 84–96 (2005)
16. Gruber, T.R.: Towards principles for the design of ontologies used for knowledge sharing. In: Guarino, N., Poli, R. (eds.) Formal Ontology in Conceptual Analysis and Knowledge Representation, Deventer, The Netherlands. Kluwer Academic Publishers, Dordrecht (1993)
17. Choi, N., Song, I.Y., Han, H.: A survey on ontology mapping. SIGMOD Rec. 35(3), 34–41 (2006)
18. Barros, A.P., Dumas, M.: The rise of web service ecosystems. IT Professional 8(5), 31–37 (2006)
19. W3C: OWL Web Ontology Language Overview (2004) W3C Recommendation (February 10, 2004)
20. IETF: The Atom Syndication Format (2005), http://tools.ietf.org/html/rfc4287
21. Treiber, M., Truong, H.L., Dustdar, S.: Semf - service evolution management framework. In: Treiber, M., Truong, H.L., Dustdar, S. (eds.) 34th Euromicro Conference on Software Engineering and Advanced Applications. SEAA 2008, pp. 329–336 (2008)
22. W3C: Xml linking language (xlink) version 1.1 (March 2008)
23. Ding, L., Zhou, L., Finin, T., Joshi, A.: How the semantic web is being used: An analysis of foaf documents, p. 113c (January 2005)
24. Golbeck, J., Rothstein, M.: Linking social networks on the web with foaf: A semantic web case study. In: Fox, D., Gomes, C.P. (eds.) AAAI, pp. 1138–1143. AAAI Press, Menlo Park (2008)
25. Canfora, G., Penta, M.D.: Testing services and service-centric systems: Challenges and opportunities. IT Professional 8(2), 10–17 (2006)
26. Demers, A., Greene, D., Hauser, C., Irish, W., Larson, J., Shenker, S., Sturgis, H., Swinehart, D., Terry, D.: Epidemic algorithms for replicated database maintenance. In: PODC 1987: Proceedings of the sixth annual ACM Symposium on Principles of distributed computing, pp. 1–12. ACM, New York (1987)
27. IBM, Microsoft: Web services inspection language (ws-inspection) 1.0 (November 2001)
28. Medjahed, B., Bouguettaya, A.: A Dynamic Foundational Architecture for Semantic Web Services. Distributed and Parallel Databases 17, 179–206 (2005)

29. Basole, R.C., Rouse, W.B.: Complexity of service value networks: conceptualization and empirical investigation. IBM Syst. J. 47(1), 53–70 (2008)
30. Allee, V.: Reconfiguring the value network. Journal of Business Strategy 21(4) (August 2000)
31. Mandelli, A.: Self-organization and new hierarchies in complex evolutionary value networks. IGI Publishing, Hershey (2004)
32. Schroth, C., Janner, T.: Web 2.0 and soa: Converging concepts enabling the internet of services. IT Professional 9(3), 36–41 (2007)
33. Shaw, M., Garlan, D.: Software architecture: perspectives on an emerging discipline. Prentice-Hall, Inc., Upper Saddle River (1996)
34. Oreizy, P., Medvidovic, N., Taylor, R.N.: Architecture-based runtime software evolution. In: ICSE 1998: Proceedings of the 20th international conference on Software engineering, Washington, DC, USA, pp. 177–186. IEEE Computer Society, Los Alamitos (1998)
35. Schall, D., Truong, H.L., Dustdar, S.: The human-provided services framework. In: CEC/EEE, pp. 149–156 (2008)