LEVERAGING THE PUBLISH-FIND-USE PARADIGM OF SOA
Supporting Enterprise Collaboration Across Organisational Boundaries

Brahmananda Sapkota and Marten van Sinderen
Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente, Drienerlolaan 5, Enschede, The Netherlands
b.sapkota@utwente.nl, m.j.vansinderen@utwente.nl

Keywords: Service Oriented Architecture, Enterprise Application Integration, Cross-organisational Collaboration, Business Requirements, Business Processes, Business-IT alignment.

Abstract: Service Oriented Architecture (SOA) has been widely recognized as an approach for flexible integration of enterprise applications across organisational boundaries based on service abstractions and Internet standards. Enterprise collaboration and application integration is driven by business requirements, which in turn are translated to business models and expressed as business processes. For example, business processes can be used to represent the coordination of several published services as well as the implementation of a composite value-added service. In this way, enterprise functions are aggregated using multi-stage business processes fulfilling the specific requirements of an enterprise. In order for enterprises to stay competitive in their respective businesses, such solutions must evolve in a timely and appropriate way in response to changes in market demands and opportunities that inevitably occur. Therefore, a mechanism is required for business-IT alignment during the complete lifecycle of SOA-based enterprise collaboration and application integration. In this paper, we discuss issues related to cross-organisational collaboration and how service-oriented principles and architectures can be applied to address these issues.

1 INTRODUCTION

Service Oriented Architecture (Erl, 2005) has been widely recognized as an approach for flexible integration of enterprise applications across organizational boundaries based on service abstractions and Internet standards. On-demand enterprise collaboration and application integration is driven by business requirements, which in turn are translated to business models and expressed as business processes. For example, business processes can be used to represent the coordination of several published services as well as the implementation of a composite value-added service. In this way, enterprise functions are aggregated using multi-stage business processes fulfilling the specific requirements of an enterprise (van Sinderen and Almeida, 2011). In order for enterprises to stay competitive in their respective businesses, such solutions must evolve in a timely and appropriate way in response to changes in market demands and opportunities that inevitably occur. Service Oriented Architecture (SOA) supports adaptation of such an evolution through the concept of service composition. The composition of services is made possible because of *service discovery* which allows to find and select the suitable services. The service discovery thus plays a central role in enterprise collaboration and application integration.

One of the fundamental requirements for the service discovery are service descriptions and service registries. In general, service descriptions are used for specifying: 1) what functionalities are offered by the service to its users (i.e., the interface definition); 2) how the service is provided (i.e., the service binding); and 3) where the service can be accessed (i.e., the service endpoint information). The service registries are used to announce the offered services and thus play crucial role for a successful on-demand enterprise collaboration and application integration. In a typical scenario, service providers publish their service descriptions to a publicly accessible service registry. The service users search over these registries and find the information needed to use the required services. This approach of publishing, finding and using the services forms the so called SOA triangular operational model which clearly separates the

Sapkota B. and van Sinderen M.
LEVERAGING THE PUBLISH-FIND-USE PARADIGM OF SOASupporting Enterprise Collaboration Across Organisational Boundaries.
DOI: 10.5220/0004458300340041
In *Proceedings of the First International Symposium on Business Modeling and Software Design (BMSD 2011)*, pages 34-41
ISBN: 978-989-8425-68-3

role of service provider, service user and service registry (van Sinderen, 2009). SOA, thus, enables flexible on-demand enterprise collaboration and application integration. Despite these sound principles of SOA, a number of practical complexities still exists, which are preventing enterprises to fully exploit the potential benefits of SOA.

An on-demand collaboration can only be achieved if 1) the participants required for the collaboration can be found and 2) the participants can communicate with each other. In the SOA based approach, the former requirement is supported by publishing the descriptions of the offered services and latter requirement is supported by defining the messages being sent. This can be realised relatively easily by using pre-meditated message structures and function libraries, if the enterprises collaborate in closed environment. If the autonomous enterprises are to collaborate, pre-meditated message structures or function libraries cannot be used. Therefore, semantics of the information provided through the service descriptions should be well defined and the service repository should contain valid service descriptions. While the initiatives around Semantic Web services have defined formalisms such as WSMO (Roman et al., 2006), OWL-S (Martin, 2004) and SAWSDL (Farrell and Lausen, 2007), to semantically define service descriptions, they cannot ensure that the published descriptions correctly reflect the offered services at the time of their discovery. Therefore, a mechanism is required for business-IT alignment during the complete lifecycle of SOA-based enterprise collaboration and application integration.

In this paper, we propose a *Summary of Services per Provider* (SSP) description based approach for service discovery and to ensure that the published descriptions are valid. An SSP description provides a means for describing the collection of services offered by a single service provider. It specifies what type of services are offered by a particular service provider. In the proposed approach, we use Web Service Definition Language (Chinnici et al., 2007) as the language for describing services and allow service providers to store them in their local repository. We provide a mechanism to generate SSP description based on the Web Service Definition Language (WSDL) documents stored at the local service registry of the service providers. These SSP descriptions are then published to the public service registry. Though the service provider still has to generate and publish these descriptions, it can be automated. Through this separation, we aim at reducing the registry updating burden at the side of the service providers. The proposed approach, therefore, follows the SOA triangular operational model except that SSP descriptions are published to the service registry instead of publishing the service descriptions.

The rest of the paper is structured as follows: Some of the highly relevant existing works are discussed in Section 2. The service discovery challenges are discussed in Section 3. The proposed solution is presented in Section 4 and its use to support cross-organisational collaboration is presented in Section 5. Finally, the work presented in this paper is concluded in Section 6 by highlighting possible future directions.

2 RELATED WORKS

The problem of guaranteeing correctness of the published service descriptions and reducing the effort required for providing such guarantees is starting to attract attention from the research communities. The work presented in (Küster and Köning-Ries, 2007) follows an approach similar to the one presented in this paper. The focus of this work is to ensure that the published service description indeed represents the concrete service. In order to support this, an estimation step followed by a single execution step is proposed. In the estimation step, additional information than that is available in the service description itself is gathered whereas in the execution step, the actual invocation of service is performed.

A preference-based selection of highly configurable web services is presented in (Lamparter et al., 2007). It focuses on defining algorithms required for finding optimal configurations while selecting the services. Unlike the work presented in this paper, their work neither considers minimisation of the extra effort required to update the service registry when service descriptions are changed nor maximising the correctness of the published service descriptions. To address the problem due to changes in service descriptions a RSS-based mechanism to announce such changes is proposed in (Treiber and Dustdar, 2007). The RSS-based approach is service provider dependent because the changing information should come from them.

Treating services from the economical point of view, (Cardoso et al., 2009) defines universal service description language. The proposed language is defined to describe both the IT and non-IT services. In the proposed language, provisions for defining dynamic information is poorly defined. The approach presented in (Truong et al., 2010) defines mechanism for identifying and reducing irrelevant information in service composition and execution. This approach

is target at increasing efficiency and correctness of the composition and execution. This approach works only after the services are discovered and does not eliminate the possibility of discovering services with incorrect information.

In contrast to many other traditional approaches, (Speiser and Harth, 2011) proposes a LinkedData based approach for integrating data providing services. Their approach is suitable for sharing data which might change over time. In comparison to the work presented in this approach, their approach requires the service providers to describe the offered services using LinkedData principles and does not support sharing of already existing service descriptions which are described using WSDL. We propose mechanisms to allow usage of WSDL while still dealing with changing and state dependent data.

A crawl based approach for collecting, annotating and classifying public Web services has been proposed in (AbuJarour et al., 2010) attempting to increase the role of service registry in service-oriented architecture by providing correct information. In their approach, publicly available web service descriptions are crawled, annotation information is gathered, Web services are annotated and classified based on this information. Through such classification, authors aim at supporting better discovery of services. In this direction, the work presented in (Obrst et al., 2010) aims at enabling rich discovery of Web services by projecting weak semantics from structural specifications. These approaches, however, cannot guarantee that the information required for service discovery is gathered.

Combination of document classification and ontology alignment schemes is proposed in (CRASSO et al., 2010) to semantically enrich Web services. Though this scheme helps in efficiently discovering required services, it does not tackle the problem of outdated service descriptions. The work presented in (da Silva et al., 2011) specifies mechanisms for runtime discovery, selection and composition of semantic services. The proposed approach supports semantic descriptions of the services but lacks support for dealing with outdated service descriptions.

3 SERVICE DISCOVERY CHALLENGES

In an open environment, it is difficult to support on-demand collaboration if the published service descriptions are either outdated or provide ambiguous or incorrect information. This difficulty escalates when service descriptions contain limited information, either because the service providers are unwilling to share all the information or because the information is state dependent (Treiber and Dustdar, 2007; Küster and Köning-Ries, 2007), i.e., the information may change as the service is invoked. This will result in a poor discovery results.

The correctness problem arises due to the fact that service registries are passive. If the functionalities of the offered services are changed, service providers should take the initiative to update the corresponding descriptions published in the service registry. In practice, the published descriptions are rarely updated. Instead of publishing the service descriptions to the service registries, they are published on the Web (Michlmayr et al., 2007). Such a practice, violates the original SOA model (i.e., the triangular operational model) and consequently undermines the role of service registries in SOA (AbuJarour et al., 2010). This shift in practice is due to the lack of efficient and elegant support to update the service registry whenever a service description is updated. The latter essentially requires an additional effort on part of the service providers. It becomes more problematic because the existing service registries emerge and disappear (Sabou and Pan, 2007) and cannot be fully relied upon for service discovery. The results of the investigation of Web services on the Web published in (Al-Masri and Mahmoud, 2008) reveals that only around 63% of the discovered Web services are in fact active. This lack of reliable service repositories makes on-demand collaboration between autonomous enterprises difficult, if not impossible.

Besides these technical difficulties, there are other reasons why public service registries have not been successful so far. One of the reasons is that a considerable amount of the published service descriptions are unusable (Treiber and Dustdar, 2007). Some of the information (e.g., the information that depends on the change in state) which might be important for discovery purposes cannot be included in the service descriptions in a simple way. In addition, some providers may not be willing to disclose information related to nun-functional properties and the quality of service parameters because of the fear of bargain or competition from other providers (Küster and Köning-Ries, 2007). This contributes to the retrieval of imprecise service descriptions leading to false positives and consequently reducing the usability and the reliability of service registries.

The above mentioned problems could be resolved if service providers are allowed to store their service description locally and provide them with a tool that extracts summary information from their repository, builds an SSP description and publishes it to the ser-

vice registry. The benefits of this approach are: 1) service providers do not have to publish all the information. It will also reduce the extra effort required for maintaining and updating the service registry. Service providers do not need to update service registry every time the service description is updated, updates are done locally. 2) service users can use these SSP descriptions to find the potential providers, and finally obtain the up-to-date information. Figure 1 shows the overall architecture of this approach.

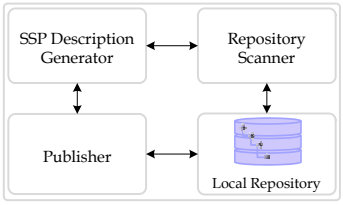

Figure 1: Overall architecture.

4 SERVICE DISCOVERY SOLUTIONS

We describe a mechanism to extract type information from the WSDL files stored in the service provider's local repository and a mechanism to publish these information as SSP description to the service repository.

4.1 Generating Type Information

One of the purposes of using type information, in the proposed approach, is to provide indication of what type of services are offered by the service providers. This kind of information serves the purpose of *guiding* service requests towards the potential service providers. The type information based approaches are expected to help in narrowing down the search space and allowing applications to deal with the ever increasing number of Web services. We extract these information from service descriptions encoded in WSDL, which is a commonly used service description language. A WSDL document is structured into four elements describing *Service*, *Bindings*, *Interface* and *Types* definitions. A *Service* definition specification specifies a collection of endpoints (i.e., URLs). The *Bindings* definition typically specifies what data formats and communication protocols to use when invoking the service. The operations, message exchange patterns and mechanisms for fault handling are specified in the *Interface* definition. The data types used in messages and faults are specified in the *Types* definition. Figure 2 shows some of these parts pictorially where as the Listing 1 shows the type and service definition parts of a WSDL document.

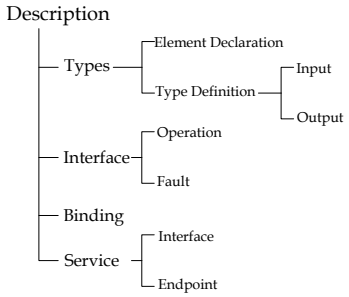

Figure 2: Pictorial representation of WSDL structure.

The data types specified in the *Types* definition essentially model the domain knowledge and thus are useful for extracting the necessary information for generating SSP description from a given WSDL document.

```
<wsdl:description
  targetNamespace="http://org.example.com/services/AvailabilityService/"
  xmlns="http://org.example.com/services/AvailabilityService/"
  xmlns:wsdl="http://www.w3.org/ns/wsdl"
  xmlns:xsd="http://www.w3.org/2001/XMLSchema">

  <wsdl:types>
   <xsd:schema targetNamespace="http://org.example.com/resources/
   AvailabilityService">
    <xsd:element name="ServiceRequest">
     <xsd:complexType>
      <xsd:sequence>
       <xsd:element name="product" type="xsd:string"/>
       <xsd:element name="date" type="xsd:string"/>
       <xsd:element name="quantity" type="xsd:float"/>
      </xsd:sequence>
     </xsd:complexType>
    </xsd:element>
    <xsd:element name="ServiceResponse" type="response"/> <xsd:element
    name="ServiceResponse" restriction="xsd:boolean"/>
    <xsd:simpleType name="response">
     <xsd:restriction base="xsd:boolean"/>
    </xsd:simpleType>
    </xsd:schema>
  </wsdl:types>
  <wsdl:interface name="AvailabilityServiceInterface">
   <wsdl:operation name="RequestOperation" pattern="http://www.w3.org/ns/
   wsdl/in-out">
    <wsdl:input element="ServiceRequest"/>
    <wsdl:output element="ServiceResponse"/>
   </wsdl:operation>
  </wsdl:interface>
</wsdl:description>
```

Listing 1: Fragment of a WSDL document.

Given a WSDL document, we extract the Types definitions and represent them as RDF (Klyne and Carroll, 2004) triples. Representing those definitions in RDF has several advantages. RDF provides an abstract model for describing resources with properties, scoping them to a particular application domain through RDF Schema (Hayes, 2004) and defining relationships between these resources. In addition, standard RDF query language SPARQL (Prud'hommeaux and Seaborne, 2007) can be used to provide flexible means to allow users to express their requests. The RDF triples are generated based on the XML

Schema to RDF Schema mappings approaches presented in (Thuy et al., 2008) and (CRASSO et al., 2010). The resulting RDF triples are shown in Listing 2.

```
<rdfs:Class rdf:ID="http://org.example.com/resources/AvailabilityService#
ServiceRequest"/>
<rdfs:Class rdf:ID="http://org.example.com/resources/AvailabilityService#
ServiceResponse"/>
<rdf:Property rdf:about="http://org.example.com/resources/AvailabilityService#
response">
  <rdfs:domain rdf:resource="http://org.example.com/resources/
  AvailabilityService#ServiceResponse"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#boolean"/>
</rdf:Property>
<rdf:Property rdf:ID="http://org.example.com/resources/AvailabilityService#
product">
  <rdfs:domain rdf:resource="http://org.example.com/resources/
  AvailabilityService#ServiceRequest"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</rdf:Property>
<rdf:Property rdf:ID="http://org.example.com/resources/AvailabilityService#date"
>
  <rdfs:domain rdf:resource="http://org.example.com/resources/
  AvailabilityService#ServiceRequest"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#string"/>
</rdf:Property>
<rdf:Property rdf:ID="http://org.example.com/resources/AvailabilityService#
quantity">
  <rdfs:domain rdf:resource="http://org.example.com/resources/
  AvailabilityService#ServiceRequest"/>
  <rdfs:range rdf:resource="http://www.w3.org/2001/XMLSchema#int"/>
</rdf:Property>
```

Listing 2: RDF representation of extracted information.

In Listing 2, each triples are encoded in the form $<s, p, o>$ where s, p, and o are called the *subject*, *predicate* and the *object* respectively. In a triple, a predicate is called the property of the triple and denotes the relationship between the subject and the object that it connects. The subject always appears at the left whereas the object appears at the right side of the predicate in the triple.

4.2 SSP Description

An SSP description models the service provider as a collection of services and enumerates all the offered services. In particular, the SSP description specifies the type of services that are offered by a particular service provider. We use the information extracted from the locally stored WSDL files to describe domain specific concepts and to avoid ambiguities between services from different application domains. Using the SSP description, functional properties of the offered services are described collectively. These descriptions provide information sufficient enough to filter out the completely irrelevant service providers.

We define SSP description as <n, e, Q>, where n is the URL of the service provider, e is the SPARQL endpoint of the local repository and Q is the collection of (<p(s_t, o_t), f>) pairs, where s_t and o_t represents the type of the subject and object connected by the predicate p whereas f represents the total number of occurrences of the predicate p together with the s_t and o_t. The frequency of occurrences of subject and object type combination of each predicate is measured to indicate the number of services from a particular domain. We included the subject and object types in the SSP description because they represent the domain and range of a predicate and hence is useful to unambiguously select the required information. The structure of the SSP description is defined as RDF graph as shown in Listing 3.

```
<?xml version="1.0"?>
<rdf:RDF xmlns:ex="http://www.example.org/ontology#"
         xmlns:ont="http://www.example.org/ns/ms-ontology#"
         xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
         xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
  <ont:Service rdf:about="http://www.example.org/provider/xyz">
    <ont:endPoint rdf:resource="http://www.example.org/services/abc"/>
    <ex:name>
      <rdf:Description>
      <ex:subjectType rdf:resource="http://www.example.org/ontology#Hotel"/>
      <ex:objectType rdf:resource="http://www.w3.org/2001/XMLSchema#string
      "/>
      <ex:count>435</ex:count>
      </rdf:Description>
    </ex:name>
    <ex:product>
    <rdf:Description>
      <ex:subjectType rdf:resource="http://www.example.org/ontology#
      Automobile"/>
      <ex:objectType rdf:resource="http://www.example.org/ontology#Wheel"/>
      <ex:count>1481</ex:count>
    </rdf:Description>
    </ex:product>
  </ont:Service>
</rdf:RDF>
```

Listing 3: An example SSP description.

The number of instances of objects and subjects in a repository is typically far larger than the number of distinct predicates. Inclusion of the objects in the SSP description is therefore likely to increase its size as close to as the size of the repository. In order to avoid such problems, the SSP description includes only the subject and object types and not their instances. This allows to select the relevant service provider based on this type information. For example, if there are a large number of printing service providers and only few of them are providing book printing services it is much more efficient to send a book printing service requests only to those that provide the book printing services. This type of SSP description will be helpful in selecting the service providers in situations where fine grained information is needed for answering the service requests.

4.3 Finding Potential Service Providers

In order to find the required service we follow two step discovery mechanism. In the first step, the discovery request is sent to the service registry from which a list of potential service providers are identified. In the second step, direct communication with the potential service providers is established to find the most appropriate service provider. The goal of the first step is to reduce the search space whereas the

second step is intended to select the service provider based on up-to-date information.

The major design goals of the proposed approach is simplicity and extensibility. The SSP description is essentially the collection of summary of the WSDL files stored in the local repository of the providers which simplifies the process of service description updates. The service providers need to update only the local repository. If the new service providers arise, they can simply publish the SSP description to the service registry.

We assume that the service providers employ the mechanisms discussed above for extracting the information needed for generating SSP descriptions. First, the service requester initiates the lookup over SSP descriptions and obtains a list of endpoints of the potential partners. These endpoints are then queried to obtain further information for selecting the potential service providers.

4.4 Maintaining Freshness

The SSP descriptions are published to the service registry and hence can still pose the same problems as with publishing the service descriptions if the domain information is changed. In order to ensure that the SSP descriptions are still up-to-date, the service repository requires a mechanism to reverse look up the service providers local repository and synchronize the information that is being provided through the SSP descriptions.

5 ENTERPRISE COLLABORATION

Let us now return to our original goal of facilitating enterprise collaboration using SOA. We previously concluded that the SOA architectural triangle with its 'publish-find-use' paradigm is in principle very convenient to enterprises to utilize distributed capabilities that may be under the control of different ownership domains. The convenience stems from the loose coupling of services - supporting flexible composition - and the external-oriented representation of services - allowing interactions between users and providers irrespective of their internal implementation. However, we also concluded that the 'publish-find' part of the triangle has practical limitations, which so far has prevented the successful uptake of public-registry/open-discovery based enterprise collaboration. The main limitations are: (a) it is hard to find and compose services based on current service descriptions, since the descriptions lack unambiguous and precise semantics; (b) it is expensive and laborious to maintain service descriptions, as the corresponding services continuously evolve and therefore the descriptions require frequent and manually managed updates; (c) trust is a hindrance for publishing service descriptions and using services discovered with public registries.

We propose to leverage the publish-find-use paradigm by using public descriptions that are automatically generated and semantically enhanced, as described in Section 4. In the following, we first show which interactions are necessary for enterprise collaboration using our approach, and subsequently discuss the potential benefits of our approach.

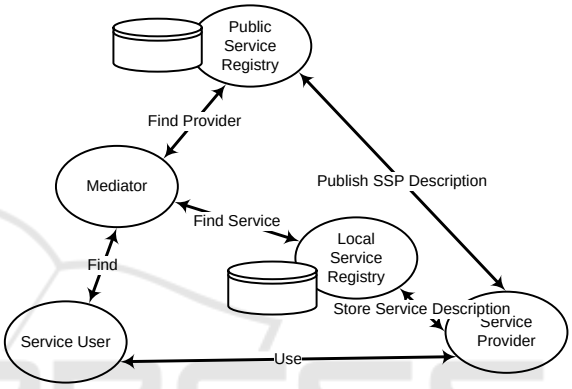

Figure 3: Proposed service discovery approach.

Figure 3 illustrates the basic interactions:

1. A business organisation acting as a service provider can publish relevant information on all its services using a single keyword-based description (SSP) at a service broker that maintains a public registry. If services offered by the business organization evolve, the SSP is re-generated or incrementally updated, depending on the nature of the change. The updated SSP can be pushed to the service broker, in similar to the original publication, where it is used to replace the old SSP. Alternatively, the service broker periodically asks the service provider for updates.
2. A business organisation looking for partners that can offer certain services can contact the service broker and find SSPs based on keyword matching. Although keyword matching can already be reasonable efficient (Obrst et al., 2010), we can further improve the recall and precision of service discovery by exploiting the RDF-based semantics of SSPs.
3. If an SSP fulfils the search criteria, the service provider is contacted via the endpoint that is part

of the SSP. Then the local repository of the service provider is used to find available services. Both step (2) and step (3) may be performed via an intermediary, making the two-step service discovery transparent to the requesting business organization. For example, the service broker may take the intermediary role. Possibly, the requested services are not available as single services or are not available from a single service provider. In that case, the services have to be offered as a bundle or must be composed. Again, an intermediary may automate or support this process (da Silva et al., 2011).

4. Once the (composed) services are found, the requesting business organization can start using them, effectively entering collaborations with one or more partner business organizations that are involved in the offering of these services.

Comparing this with the enterprise collaboration through public UDDI registries, we observe the following benefits:

- The SSP is based on extracting keywords from WSDL type definitions, and represents these keywords and their relationships with RDF. In this way, the semantic properties of keywords can be captured (CRASSO et al., 2010; Thuy et al., 2008). This addresses the limitation (a) mentioned above.
- Since the extraction is automatic, the burden for service providers to update descriptions is dramatically lowered. Moreover, if the service broker is able to poll for updates, the problem of 'disappearing' business organizations and 'ghost' services can be tackled. If a business organization no longer supports its previously published services, e.g. because it no longer exists, a poll for updates by the service broker gets no reaction and the service broker can decide to remove the SSP from its registry. This addresses the limitation (b) mentioned above.
- The two-step service discovery approach has the advantage that it first determines the services providers that offer potentially relevant services, and then limits the search for services to those of the selected service providers. Although we still have to confirm this with experiments, we believe that this approach has a better scalability than one-level semantic search. Furthermore, by favoring services from the same or a few providers, it is more likely that these services are defined and implemented in a consistent way, making search and composition easier and more efficient (Forestiero et al., 2010).
- In order to address limitation (c) mentioned above, the local registry of a service provider may be enhanced in two ways. First, the service provider may monitor who wants to access the local registry, and expose information on its services depending on some trust classification scheme (e.g., based on previous collaborations). Secondly, the service provider may provide additional information through the local registry, which facilitates the (non-) selection of services. For example, non-functional properties based on resource availability or historical data may be published, including information on trust, security or privacy aspects.
- The implications for existing standards, most notably UDDI, is minimal. Most of the interactions described above can be supported with UDDI as is.

6 CONCLUSIONS

Our research demonstrates how SSP descriptions allows us to reduce maintainability cost at the service providers side while still providing the relevant information required for service discovery. This type of approach has the ability to guarantee better service results due to the separation of abridged service descriptions and the actual detailed descriptions. Current approaches either do not provide adequate support for publishing accurate information of the offered services or the offered solutions are too restrictive in terms of cost and time required for maintaining the published descriptions. This is mainly because the service descriptions are valid only at the time they are created and subject to frequent change depending on changes in market trends.

ACKNOWLEDGEMENTS

This material is based upon works jointly supported by the IOP GenCom U-Care project (http://ucare.ewi.utwente.nl) sponsored by the Dutch Ministry of Economic Affairs under contract IGC0816 and by the DySCoTec project sponsored by the Centre for Telematics and Information Technology (CTIT), University of Twente, The Netherlands.

REFERENCES

AbuJarour, M., Naumann, F., and Craculeac, M. (2010). Collecting, Annotating, and Classifying Public Web Services. In *Proc. of International Conference on On the Move to Meaningful Internet Systems*, pages 256–272.

Al-Masri, E. and Mahmoud, Q. H. (2008). Investigating Web Services on the World Wide Web. In *Proc. of the World Wide Web Conference*, pages 759–804.

Cardoso, J., Winkler, M., and Voigt, K. (2009). A Service Description Language for the Internet of Services. In *Proc. of the International Symposium on Services Science*.

Chinnici, R., Moreau, J.-J., Ryman, A., and Weerawarana, S. (2007). *Web Services Description Language (WSDL) Version 2.0*.

CRASSO, M., ZUNINO, A., and CAMPO, M. (2010). Combining Document Classification and Ontology Alignment for Semantically Enriching Web Services. *New Generation Computing*, 28:371–403.

da Silva, E. G., Pires, L. F., and van Sinderen, M. (2011). Towards runtime discovery, selection and composition of semantic services. *Computer Communications*, 34(2):159–168.

Erl, T. (2005). *Service-Oriented Architecture Concepts, Technology, and Design*. Prentice Hall Professional Technical Reference.

Farrell, J. and Lausen, H. (2007). *Semantic Annotations for WSDL and XML Schema*.

Forestiero, A., Mastroianni, C., Papuzzo, G., and Spezzano, G. (2010). A Proximity-Based Self-Organizing Framework for Service Composition and Discovery. In *Proc. of the 10th IEEE/ACM International Conference on Cluster, Cloud and Grid Computing*, pages 428–437.

Hayes, P., editor (2004). *RDF Semantics*. W3C Recommendation.

Klyne, G. and Carroll, J. J., editors (2004). *Resource Description Framework: Concepts and Abstract Syntax*. W3C Recommendation.

Küster, U. and Köning-Ries, B. (2007). Supporting Dynamics in Service Descriptions - The Key to Automatic Service Usage. In *Proc. of the 5th International Conference on Service-Oriented Computing*, pages 220–232.

Lamparter, S., Ankolekar, A., and Grimm, S. (2007). Preference-based Selection of Highly Configurable Web Services. In *Proc. of the 16th International Conference on World Wide Web*, pages 1013–1022.

Martin, D., editor (2004). *OWL-S: Semantic Markup for Web Services*. W3C Member Submission.

Michlmayr, A., Rosenberg, F., Platzer, C., Treiber, M., and Dustdar, S. (2007). Towards Recovering the Broken SOA Triangle: A Software Engineering Perspective. In *Proc. of the 2nd International Workshop on Service Oriented Software Engineering*, pages 22–28.

Obrst, L., McCandless, D., and Bankston, M. (2010). Enabling Rich Discovery of Web Services by Projecting Weak Semantics from Structural Specifications. In *Proc. of Semantic Technology for Intelligence, Defense, and Security*.

Prud'hommeaux, E. and Seaborne, A., editors (2007). *SPARQL Query Language for RDF*. W3C Candidate Recommendation.

Roman, D., Lausen, H., and Keller, U., editors (2006). *Web Service Modeling Ontology (WSMO)*. WSMO Working Group.

Sabou, M. and Pan, J. (2007). Towards semantically enhanced Web service repositories. *Web Semantics: Science, Services and Agents on the World Wide Web*, 5(2):142–150.

Speiser, S. and Harth, A. (2011). Integrating Linked Data and Services with LIDS. In *Proc. of the 8th Extended Semantic Web Conference*.

Thuy, P. T. T., Lee, Y.-K., Lee, S., and Jeong, B.-S. (2008). Exploiting XML Schema for Interpreting XML Documents as RDF. In *Proc. of the 2008 IEEE International Conference on Services Computing*, pages 555–558.

Treiber, M. and Dustdar, S. (2007). Active Web Service Registries. *IEEE Internet Computing*, 11(5):66–71.

Truong, H.-L., Comerio, M., Maurino, A., Dustdar, S., Paoli, F. D., and Panziera, L. (2010). On Identifying and Reducing Irrelevant Information in Service Composition and Execution. In *Proc. of the International Conference on Web Information Systems Engineering*, pages 52–66.

van Sinderen, M. (2009). From Service-Oriented Architecture to Service-Oriented Enterprise. In *Proc. of the Third International Workshop on Enterprise Systems*, pages 3–16.

van Sinderen, M. and Almeida, J. P. A. (2011). Empowering Enterprises through Next-Generation Enterprise Computing. *Enterprise Information Systems*, 5(1):1–8.