Service Oriented Collaborative Network Architecture

Mahdi Sargolzaei(✉) and Hamideh Afsarmanesh(✉)

Springer-Verlag, Computer Science Editorial, Tiergartenstr. 17, 69121 Heidelberg, Germany
{M.sargolzaei1,H.afsarmanesh}@uva.nl

Abstract. A service-oriented collaborative network (SOCN) supports collaboration among a network of organizations through their shared business services. SOCN in comparison with traditional collaborative networks, promotes and simplifies reusability and interconnection of shared software services, in a distributed manner. Our work contributes to comprehensive support of software service oriented collaboration among networked organizations enabling semi-automated service discovery, selection, and composition in collaborative environments. With the help of an organizational monitoring tool, we improve the accuracy of claimed characteristics based on non-functional criteria of services. The reference framework and implementation architecture are defined in this paper to support implementing SOCN.

Keywords: Service oriented architecture (SOA) · Business services · Service composition · Virtual organization

1 Introduction

In today's economy, cooperations between organizations tend to change from traditional static supply chains to/dynamic organization networks [24]. In general, making relationships with business partners for the organizations brings forth lower cost, higher quality service/product, larger service/product portfolio, faster delivery, and more agility. The pace by which these changes need to occur results in high demands on supporting collaboration in a network of organizations, i.e. on establishing collaborative networks (CNs). On the other hands, organizations constantly search for innovative ways to improve their business processes and to enrich the collaborations of their distributed workers [10]. Therefore in this paper, the promising paradigm of Service Oriented Architecture (SOA) is investigated and applied to the enhancement of collaborative networks. Besides proposing the use of SOA in the construction of CNs, we extend the generic model of SOA as a reference framework to better support them. An implementation architecture is then needed to be elaborated based on the findings identified from the reference framework.

This paper specifically aims to introduce this new framework supporting collaboration among a network of organizations through their shared business services (BSs). Applying this framework results facilitates SOCN, which in comparison with traditional collaborative networks, promotes and simplifies reusability and interconnection of

L.M. Camarinha-Matos et al. (Eds.): PRO-VE 2017, IFIP AICT 506, pp. 381–394, 2017.
DOI: 10.1007/978-3-319-65151-4_35

shared software services, in a distributed manner. Furthermore, a set of sub-systems are designed and interconnected within an implementation architecture, to support all SOCN's needed functionality.

Service oriented architecture (SOA) is gaining more and more attention in business and industry since it offers new, agile and flexible ways of supporting the activities inter- and intra-organizations. These kinds of organizations propounds the idea of service orientated collaborative networks (SOCNs). However, the current implementations of the SOA approaches often do not deliver the expected advantages [24]. Several major problems in this area can be identified. First, a complete and clear understanding of the notion of SOCN is missing. As a consequence, it is unclear what functionality should be implemented to realize such system [2]. Second, an appropriate framework for specifying business services is needed to supplement insufficient information for the business services concerning various aspects of the respective services [24] and [6]. Third, an enhanced service discovery mechanism is needful in order to give more flexibility and efficiency to search for services in more impressive ways rather than only through general keyword-based delineation of required services [1] and [6]. Finally, to fully leverage the opportunities provided by the service oriented architecture within organizations, integration of existing services must be simplified [9, 23] and [2]. As a consequence, approaches for automated service composition, as well as the execution of the integrated services are needed. The above problems present the motivation of this paper.

The paper starts with a brief description of some related works in Sect. 2. In Sect. 3, a short introduction of the Service Oriented Architecture is represented. In Sect. 4 the role of SOA in today's organizations is discussed. An extended model of SOA paradigm is addressed in Sect. 5 to identify the basic requirements to support SOCN. Here a reference framework and architecture to facilitate assisting the implementation of our proposed model is presented thereafter, identifying and brie y describing its main components. We conclude the paper in Sect. 6.

2 Related Work

Multiple approaches are developed by research community in the area of service oriented computing (SOC). However, only a few address the comprehensive design of a framework that can encompass all aspects required to efficiently support semi-automated service discovery and composition within SOA-based virtual organizations.

Specification of services is the first base step of SOC that can improve discovery and composition of services. Nowadays, research on web services constitutes the base in Service-Oriented Architecture research [1], and has been widely accepted by service industry. One key point for the success of web service technology is the employment of XML-based standards, such as SOAP and WSDL for communicating and self-describing it [4]. These standards enable web services to become independent of the programming language, operating system, and hardware [9]. It therefore supports communication in heterogeneous environments, and is ideally suited to provide dynamic information and functionalities over the CNs. However, applying WSDL as the most adopted base standard for web Service description is limited to self-describing of the

structure of the messages and operations, and not supporting the conceptual description or the capability of the service. This limitation, which is known as lack of semantics and behavior in describing web services [2], consequently required human intervention to ensure valid and befitting use of the services. Currently, the most promising research works in the area of software service specification are semantic service description frameworks, such as OWL-S, WSMF, and WSDL-S that provide machine understandable semantic description of services in order to enable automatic service discovery and composition. Behavior of the services represents the configuration of stateful web services. In fact, the behavior specification of a service indicates the valid sequence of the operations' invocations in that service, which is also absent in WSDL. Thus, lack of semantics and behavior representation is a major drawback of WSDL, and consequently become a barrier in achieving automatic or semi-automatic service discovery, composition and execution [2].

A handful of researchers have investigated the problem of business process discovery and reuse based on process similarity, and discovery of processes based on search through repositories. However, as a consequence of insufficient information in the current service specifications, precise matchmaking required to locate a demanded service is also not supported [21]. For instance, WSXplorer [25] presents a method to retrieve desired web service operations of a given textual description. The method uses the concept of tree edit distance to match similar operations. Meanwhile, a few other algorithms are proposed for measuring and grouping similar operations. The proposed algorithms suggested for measuring and grouping similar operations catche not only the structures, but also their semantic information. Although, they still ignore behavioral and non-functional properties of services. VU+ [13] is another approach that supports semantic discovery of web services. It also performs a QoS enabled ranking of web services based on their quality compliance. The framework could be used either as a centralized discovery component or as a decentralized repository system. However, VU + is also suffers from the lack of supporting behavioral matching of services.

Service composition also has received much interest for supporting collaboration among enterprise applications [26]. Many research projects have studied this issue as the most promising choice to implement service oriented architecture and its strategic objectives. Some earlier approaches, such as the SWORD project [19] give a simple mechanism for finding and combining web services, without considering the existing complexities such as coordination among the component services. SWORD uses an expert system to check whether or not a desired composite service can be realized using existing services. If so, it designs a plan for that service composition. SWORD does not benefit from service-description standards (e.g. WSDL, SOAP, RDF and etc.), instead it uses the Entity-Relationship (ER) model to specify Web services. SWORD focuses more on the data integration and no coordination has been addressed. Some other research works on service composition rely more on QoS concerns rather than service interoperability and coordination, in order to optimize the quality of service composition. The eFlow project, WSQoSX, and UPWSR are some instances of this kind of research efforts. BPEL4WS (Business Process Execution Language for Web Services) [12], or BPEL in short, is an XML-based language for web service composition through orchestration that also supports specification of processes that involve operations

provided by one or several web services. Furthermore, BPEL4WS draws upon concepts developed in the area of workflow management. When compared to languages supported by existing workflow systems and to related standards (e.g. WSCI), BPEL4WS is relatively more expressive. BPEL however does not provide a full graphical notation, and thus needs to be combined with the BPMN to fully support service interactions [12]. Moreover, according to the principal of "separation of concerns" as a main concern in software engineering, employing the BPMN and BPEL simultaneously is not recommended. Doedt et al. [5] argue that while a business expert is aware of the BPs in an organization, he/she does not often know how to implement a service. Similarly, while the programmer knows how to implement a described service, he/she is not properly aware of the nature of processes running at the organization. It is therefore advised to deal with these two perspectives separately through different notations and standards for business process modeling and coordination. With this strategy, we have used Reo [11] as a graphical coordination language, only to implement the coordination aspects of web services.

3 Service Oriented Architecture

Recently, Service Oriented Architecture (SOA) has become a well-known and somewhat imprecise term. For example, an organizational methodology to design systems, an IT infrastructure in business, and a structure for increasing the efficiency of IT are definitions of SOA from three different points of view.

As a preliminary definition, SOA is a loosely-coupled architecture designed to meet the business requirements of the organization. It means that the dependency among services is minimized, while they only require to be aware of each other. Being open, extensible, federated, and composable are the characteristics of SOA, formally defined by Erl in [7], promoting service-orientation in enterprises. In his point of view, services in SOA are autonomous, QoS-capable, vendor diverse, interoperable, discoverable, and potentially reusable, which are implemented as Web services.

Organization for the Advancement of Structured Information Standards (OA-SIS) also represents a holistic definition of SOA as below:

"Paradigm for organizing and utilizing distributed capabilities that may be under the control of different ownership domains. It provides a uniform means to offer, discover, interact with and use capabilities to produce desired effects consistent with measurable preconditions and expectations" [15].

Indeed, Service Oriented Architecture is a programming paradigm that uses "services" as the constructs of distributed business applications to support reusability and agility. Services in this architecture are autonomous and platform-independent computational entities that can represent the steps of a business process and communicate with each other [17]. These services can be described, published, discovered, and dynamically integrated in order to developing massively distributed, interoperable, and evolvable applications. Each service can perform some functions that are able to execute either simple requests or complex business processes through peer-to-peer relationships between service clients and providers.

For a long time, increasing the level of abstraction in programing has been a main motivating goal in software engineering. Therefore, we have witnessed evolutions in programming approaches from sequential programming, respectively to procedural programming, object-oriented programming, component-based programming, and finally service oriented programming. Figure 1 shows the time-line of increasing abstraction level in programming paradigm from procedural methodology to SOA. Service oriented programming holds the promise of moving beyond the simple exchange of information and remotely invocation of methods on objects, to the concept of encapsulating data and application to services and passing messages between them. An important economic bene t of this shift in programing paradigm is that it improves the effectiveness of software development activities and enables enterprises to bring their new applications to the market more rapidly and cost-effectively than ever before, by developing composite services from the existing component applications [16] and [17].

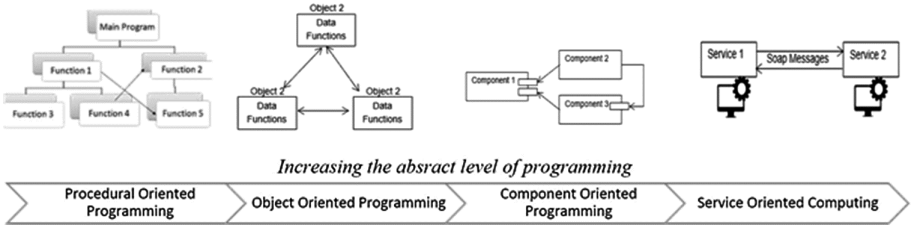

Fig. 1. The history line of SOA

Currently, web services are the most promising technology that implements the concept of SOA, and provides the basis for the development and execution of business processes that are distributed over the Internet. A web service is defined as a self-contained, modular, loosely coupled, reusable software application that can be described, published, discovered, and invoked over the World Wide Web [18, 20] and [3].

4 SOA-Based Organizations

Nowadays, SOA is a very promising methodology for supporting the business processes of organizations. Applying the service oriented architecture paradigm and technologies in organizations leads to reduced complexity and costs, reusing business functionality and operations, and finally increasing flexibility and efficiency [17]. These advantages allow organizations to adapt more readily to needed changes; therefore, it is expected that the SOA paradigm improves the e efficiency of organizations more than previous approaches and technologies.

SOA offers new and flexible ways to develop tools for supporting the activities inter- and intra-organizations. Thus, two types of usages, Intra-Organizational and Inter-Organizational platforms, can utilize service oriented computing (SOC) tools. In a large enterprise/organization, single departments can share their services stored in a

repository, and offer them to other departments to be accessed and applied to their own activities, and even to use them for purposes different than what the business service was originally built. So, this usage of SOC tool could be seen as Intra-Organizational application of SOA. The other type of using SOA in organizations is in Inter-Organizational platform, which can be established among a variety of organizations and SMEs as stakeholders that form a VO. Each VO partner should announce their services in a collaboration space, i.e. a directory of shared services, to be identifiable and accessible for other VO partners. Figure 2 shows these two kinds of usages of SOA-based organizations.

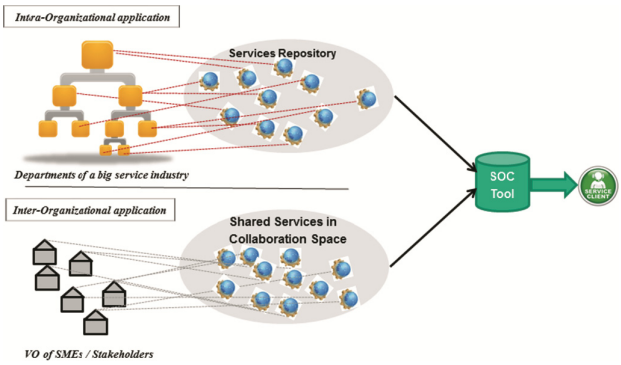

Fig. 2. Two kinds of SOA-based organizational applications

5 Towards Service Oriented Collaborative Networks

We believe that applying SOA paradigm to collaborative networks ensures a high level of abstraction for data and operations in the form of business services. Thanks to the higher abstraction level, the ease of integration with functional capabilities (i.e. services) has gradually gained power.

Business services (BSs) constitute the base construction element in service oriented virtual organizations (VOs). Each BS shared at the VO is represented as a set of Business Processes (BPs), while each BP involves the invocation of one granular software service. Therefore, designed BSs may be specified either as atomic services or composite services, whereas each composite service in turn is composed of several other atomic services as its constituent. In VOs, there are usually two kinds of business services, including the manual services and the software services. In our proposed framework, we only address software services, which correspond to their defined BPs. For manual services, if desired, we can define a simple software service to include two basic operations: Start and Stop. Only through this specification, manual tasks cab also be specified as software services in the framework, otherwise they are outside the scope of this paper.

To support the challenging task of business service composition in VOs, such software services need to be both discoverable and integrable, as it is considered in the

designed framework. We need to study the life cycle of BSs in VOs to explore the needs and challenges of establishing a service oriented collaborative network.

Business services are naturally dynamic; therefore, the supporting tools for their development and deployment are needed to be used at different stages of the Service Life Cycle (SLC). At the macroscopic level, the life cycle of business services consists of four phases, i.e. Design, Construction, Operation, and Innovation, as illustrated in Fig. 3. The need for each phase of the SLC is brie y described below.

- **1st SLC phase- Design:** This stage deal with strategic planning and rough design specification of business services. In this stage, we need to identify the required services (manual task or software services), their goals and functionalities.
- **2nd SLC phase- Construction:** This stage deals with the logistics, construction, and procurement of business services. Therefore, a number of functionalities (operations of a service) need to be implemented in this stage to support the configuration and establishment of the needed business services, for the purpose of service implementation. Moreover, this stage encompasses a fully detailed specification of business services, such as the interface of services (WSDL documents).
- **3rd SLC phase- Operation:** It is considered as the long operation phase of existing business services. Therefore, it encompasses a large number of functionalities related to the operation, management, deployment, announcement and delivery of the services. The utilization of offered services, such as service registry, discovery and execution continue heavily in this stage. Moreover, in this stage of SLC, the quality of services is measured in order to make service level agreements (SLAs).
- **4th SLC phase- innovation:** Finally, the innovation or design of new business services would be necessary for solving some emerged problems, or service enhancement/adaptation. In this stage of SLC, service designers can design new value added services through adaptation of the service interfaces, or composing some existing services. Moreover, it is possible to identify exactly the same services (functionality) and replace them with the optimized alternative service (non-functionality).

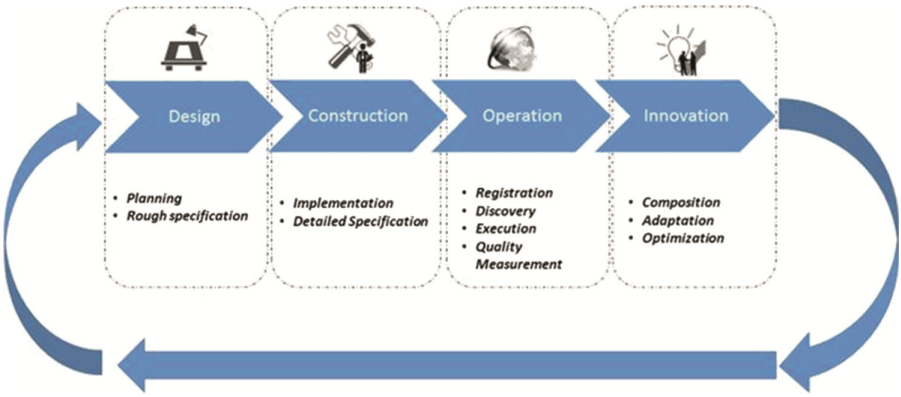

Fig. 3. Service life cycle's phases

Figure 4 shows the UML sequence diagram for the main operations involved with business services in SLC. At first, a Service Developer must add its provided services to the Service Registry. Then, a Service Client can ask Service Discovery for his/her desired business services, and consequently receive a WSDL URI as the response. The Service Discovery should send a query and receive corresponding results from the Service Registry to provide response for the Service Client. Moreover, for the purpose of providing composed business services, a Service Integrator can retrieve two or more business services from the Service Registry as the "Constituent Services", and then integrate them as a composite or composed business service, and finally publish it in the Service Registry.

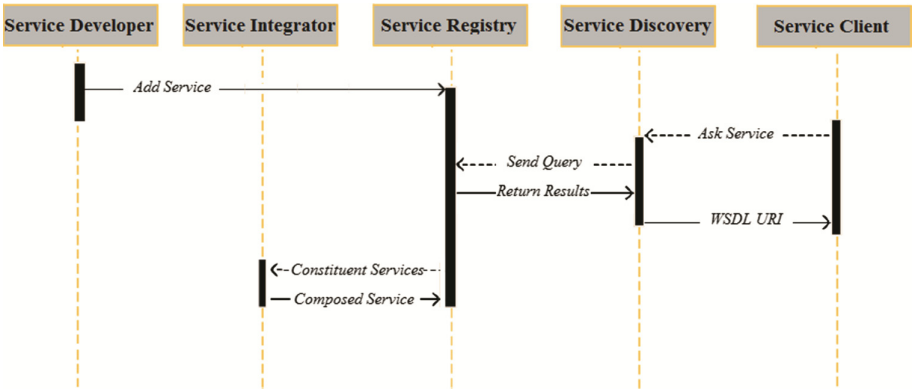

Fig. 4. The UML sequence diagram for SLC

Considering the discussion above, we have identified a set of functional and non-functional requirements to establish a framework for business service inter-operation in CNs. Note that SMEs (VO members) in the VBE are fully independent and autonomous; therefore, the lack of uniformity in full and formal definition of implemented software is quite challenging. Moreover, functionality and interactions within each component service (we have called it behavior) are not addressed in the former works, which raise some challenges in semi-automated service discovery and composition. The non-functional requirements mostly address the specifications of needed meta-data for business services, as follows:

- Unified formalism of Syntax, semantics and quality criteria of services.
- Formalization of service behavior, which tries to model the externally observable behavior of business services.
- Other non-functional requirements for software systems, e.g. security, trustworthiness and scalability.

Besides the non-functional requirements, there are also some functional requirements for business services as described below, which need to be supported:

- Service specification/registration tool to store and index the specified meta-data for services.

- Effective service discovery to search among registered services, based on their specifications.
- Support of Bundled/composite services to make a bundle of atomic business services as an integrated composed business service.

We address these functional and non-functional requirements in design of our proposed reference framework for service oriented collaborative networks (see next section), except some non-functional requirements, such as security and authorization that are out of the scope of this paper.

5.1 Reference Framework

Figure 5 shows the traditional view of service oriented architecture, which consists of three major tasks: Service Consumption, Service Provision and Service Registry. In order to customize the traditional view of the SOA for our purpose, at first we should add a sub-task, namely "Service Design & Implementation" besides the "Service Provision". Figure 6 shows this variation of the SOA paradigm. In fact, we have extended the traditional view of SOA by separating design/implementation from the provision task, which involves providing common VO business service meta-data and selection criteria. The meta-data specifies capability of a business services. Moreover, VBE that facilitates collaborative environment for business services, monitor and adjust the non-functional values claimed for such services.

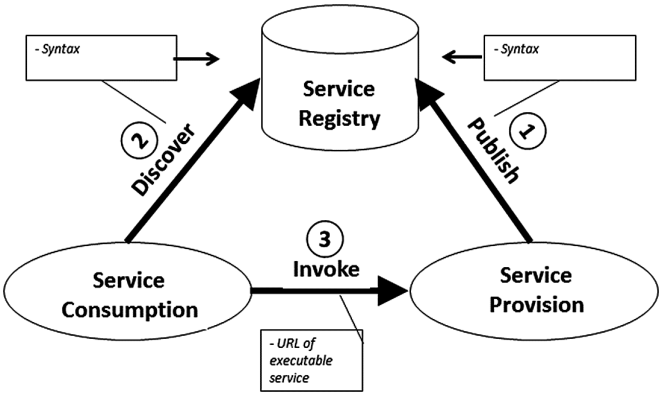

Fig. 5. Traditional view of Service oriented architecture

Figure 7 shows the second variation of the SOA that addresses service composition for SOCN precisely. Similar to the atomic services (see Fig. 6), composite service provision is also separated from the composite service design and implementation. To design composite services, at first "Service Integrator" should discover (from registry) the constituent services that form the composite service. After that, one proxy should be automatically generated for each selected constituent service to assist end users in execution of the services. In fact, the generated

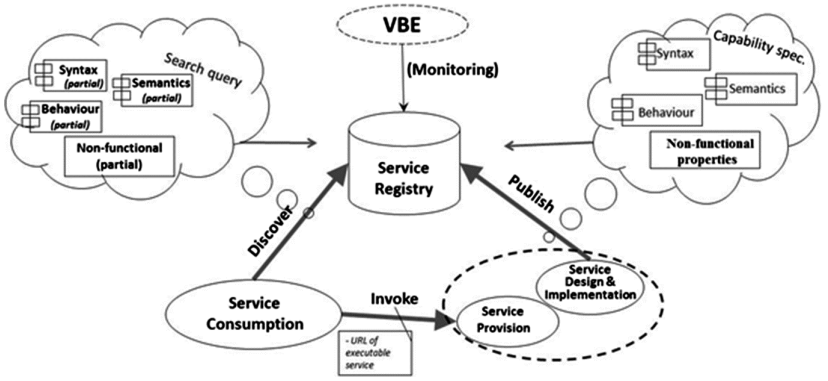

Fig. 6. The first variation of SOA needed for SOCN

proxies are required to support automated invocation and data exchanging among services. Moreover, an integrated service designer/implementer is needed to interconnect (coordinate) the selected services, and then de ne a full specification of the integrated services as a new (composite) business service. Finally, the provided composite services should be published in the "Service Registry". Thus, such services can be discovered in the SOA triangle like atomic services.

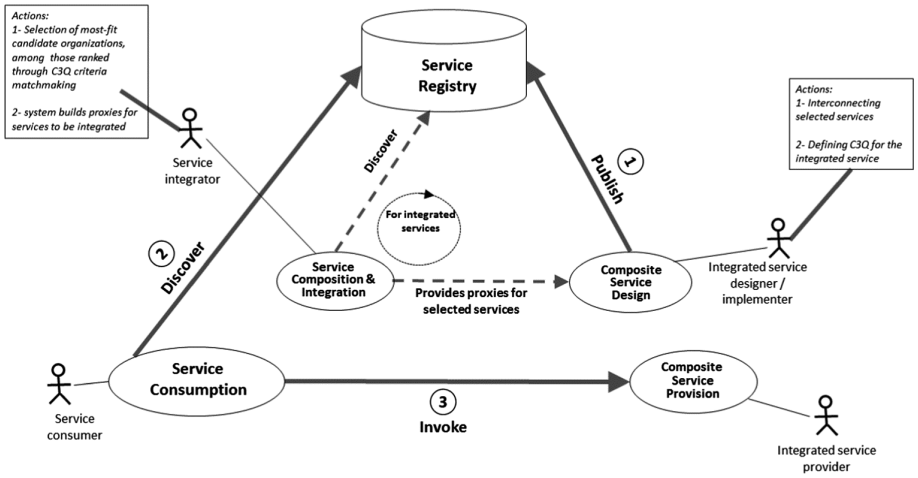

Fig. 7. The second variation of the SOA addressing service composition in SOCN.

5.2 Implementation Architecture

Figure 8 illustrates an implementation architecture capturing the needed elements for establishing service oriented computing in VOs. This architecture is designed in order to identify significant entities and relationships between them, for the development of the extended SOA model (represented in Fig. 7). Note that the proposed architecture is

not directly tied to any standards, tools or other concrete implementation technologies. This architecture conceptually is composed of three software modules, including Specification Module, Discovery Module, and Composition Module, as further explained below.

- **Specification Module** is involved with software services that are offered by different members/stakeholders of the VO in the role of service providers. The shared services are published in a service registry or directory, such as the UDDI [3], according to specific Operational Level Agreement (OLA) [6]. An OLA is agreed among the VO stakeholders, to describe the responsibilities of each VO member/stakeholder toward the specified composite services. OLAs are also supported by Service Level Agreements (SLAs) among web services [9]. At the service binding time, SLA defines the agreement between the client and the service provider to represent the expected quality and performance of the web service. Our approach to service quality assessment is rooted in [15], which identifies the level of VO partners' trustworthiness through monitoring their behaviors. Based on this approach, all agreements in OLA and SLA are considered as promises exchanged among the VO members. Using VO Supervisory Assisting Tool (VOSAT) [15], at any point in time, the trust level of a VO member would be reflected on its claims about different QoS properties of its provided services, as well as on its feedbacks related to its consumed services.
  The function that is labeled as "Agreement Management" manages all tasks related to the service agreements, and keep the results in the Service Registry. The other function of this module named "Software Service Specification", which presents the triple meta-data content as the means of accurate formalization for every atomic software service. Every composite service, is then specified by a set of meta-data, as the concrete formalization of its associated atomic services. To support machine-to-machine service interoperation in a VO, our proposed meta-data reflects the three aspects of each service, namely its syntax, semantics, and behavior, where they facilitate discovery and composition of services. Moreover, these specifications are deployed to execute the services in an automated way. All of the specifications will be registered in the "Service Registry".
  We have introduced XWSDL, as our extension to WSDL that can support all information aspects needed for our semi-outdated service discovery and composition. We have also implemented a GUI for the behavioral specification of web services. As Fig. 9 shows, the GUI accepts one WSDL file as its input and then depict its behavior in the form of constraint automata. Figure 9 shows a snapshot of this GUI.
- **Discovery Module** of the proposed architecture represents the needed tasks for service discovery and selection of the existing web services in the VO. Automated and successful application of search services at this module needs to the functional meta-data (i.e. syntax, semantics, and behavior of services) provided in the Specification Module. The Search function returns a list of alternative services from the Service Registry that can be matched with the service query (based on the functional properties of services). After that, the Service Selection function chooses one of the service alternatives that optimize non-functional properties of services.
  We have implemented a tool for the proposed discovery module of the architecture. It is able to rank the shared service descriptions in VBE, according to the matched

similarity score for each service description against the service desired by a user. More details about this tool has represented in [21].

- **Composition Module** of the implementation architecture involves the functions needed for service composition, to allow service integrators bundle several shared services in the VO, and offer a new composite service. An efficient service composition in this module requires not only the rich meta-data captured in the Service Registry, but also the coordination model that reflects the demanded interactions among the constituent services that form a composed service. The information of the constituent services as well as the orchestration model should be kept in the Service Registry.
  We also implemented ProxCG as a proxy generator to realize the composition module of the architecture. The proposed approach is based on Reo, which is a graphical and domain specific language for coordination of services. The details of ProXCG can be found in [11]. Figure 10 shows a screenshot of the ProxCG, which is implemented within a tool that works with Reo, so-called ETC [11].

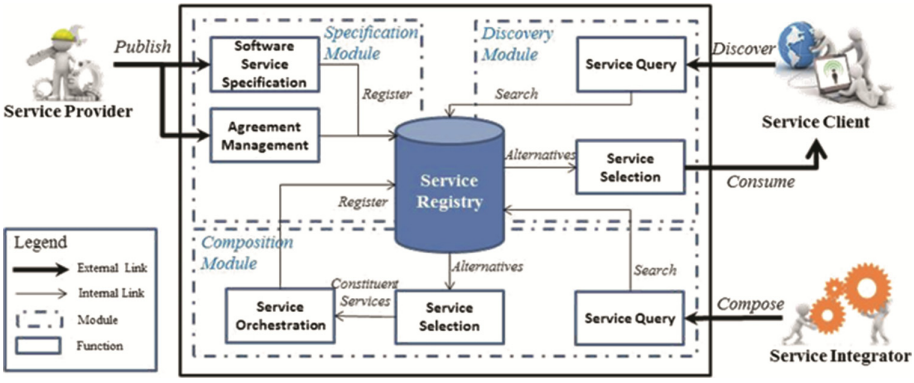

Fig. 8. Implementation Architecture of SOCN.

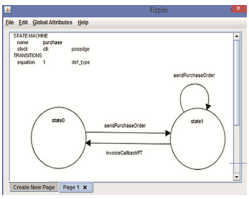

Fig. 9. A snapshot of the GUI

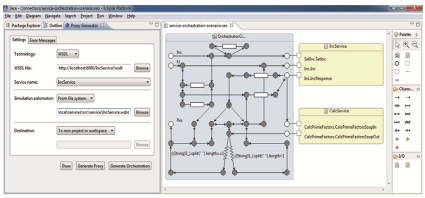

Fig. 10. A snapshot of the ProxCG.

6 Conclusion

Emerging developments under the umbrella of "Future Internet" and particularly on web services, highlight SOA-based paradigms and approaches to support service oriented collaborative networks. Software services, e.g. the web services, and SOA paradigm

provide rapid, cost-effective and standard-based means to improve service interoperability and collaboration in CNs. Although the research area of collaborative networks is active, the higher-level abstraction of the activities that simplify the collaboration among members using SOA is still lacking. The goal of this paper is to address a comprehensive framework in order to support a semi-automated service discovery and compositions in VOs. With this in mind, we propose a reference framework and an implementation architecture that allows us to efficiently support service-oriented collaborative networks. The implementation of the proposed framework consists of three software modules that realize functional and non-functional requirements of SOCN addressed in this paper.

References

1. Afsarmanesh, H., Sargolzaei, M., Shadi, M.: A framework for automated service composition in collaborative networks. In: Camarinha-Matos, Luis M., Xu, L., Afsarmanesh, H. (eds.) PRO-VE 2012. IAICT, vol. 380, pp. 63–73. Springer, Heidelberg (2012). doi: 10.1007/978-3-642-32775-9_7
2. Afsarmanesh, H., Sargolzaei, M., Shadi, M.: Semi-automated software service integration in virtual organisations. Enterp. Inf. Syst. **9**(5–6), 528–555 (2015)
3. Curbera, F., Duftler, M., Khalaf, R., Nagy, W., Mukhi, N., Weerawarana, S.: Un-raveling the web services web: an introduction to SOAP, WSDL, and UDDI. IEEE Int. Comput. **6**(2), 86–93 (2002)
4. Dhara, K., Dharmala, M., Sharma, C.: A Survey Paper on Service Oriented Architecture Approach and Modern Web Services (2015). http://opus.govst.edu/capstones/157
5. Doedt, M., Stefien, B.: An evaluation of service integration approaches of business process management systems. In: 2012 35th Annual IEEE Software Engineering Workshop (SEW), pp. 158–167. IEEE (2012)
6. Du, X.: Semantic service description framework for efficient service discovery and composition. Ph.D. thesis, Durham University (2009)
7. Erl, T.: Service-Oriented Architecture: Concepts, Technology, and Design. Pearson Education, Mumbai (2005)
8. Guidara, I., Chaari, T., Fakhfakh, K., Jmaiel, M.: A comprehensive survey on intra and inter organizational agreements. In: 2012 IEEE 21st International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pp. 411–416. IEEE (2012)
9. Huma, Z., Gerth, C., Engels, G., Juwig, O.: Automated service discovery and composition for on-the-fly SOAs. Technical report, TR-RI-13-333, University of Paderborn, Germany (2013). http://is.uni-paderborn.de/uploads/txsibibtex/tr-ri-13-333.pdf
10. Jerstad, I., Dustdar, S., Thanh, D.V.: A service oriented architecture framework for collaborative services. In: 2005 14th IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprise, pp. 121–125. IEEE (2005)
11. Jongmans, S.S.T., Santini, F., Sargolzaei, M., Arbab, F., Afsarmanesh, H.: Orchestrating web services using Reo: from circuits and behaviors to automatically generated code. SOCA **8**(4), 277–297 (2014)
12. Kopp, O., Leymann, F., Wagner, S.: Modeling choreographies: BPMN 2.0 versus BPEL-based approaches. In: EMISA, pp. 225–230 (2011)
13. Le-Hung, V., et al.: A search engine for QoS-enabled discovery of semantic web services. Int. J. Bus. Process Integr. Manag. **1**(4), 244–255 (2006)

14. Ludwig, H., Keller, A., Dan, A., King, R.P., Franck, R.: Web service level agreement (WSLA) language specification. IBM Corporation, pp. 815–824 (2003)
15. MacKenzie, C.M., Laskey, K., McCabe, F., Brown, P.F., Metz, R., Hamilton, B.A.: Reference model for service oriented architecture 1.0. OASIS standard 12, p. 18 (2006)
16. Mallayya, D., Ramachandran, B., Viswanathan, S.: An automatic web service composition framework using QoS-based web service ranking algorithm. Sci. World J. **2015** (2015)
17. Papazoglou, M.P., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: a research roadmap. Int. J. Coop. Inf. Syst. **17**(02), 223–255 (2008)
18. Petritsch, H.: Service-Oriented Architecture (SOA) vs. Component Based Architecture. Vienna University of Technology, Vienna (2006)
19. Ponnekanti, S., Fox, A.: SWORD: a developer toolkit for web service composition. In: Proceedings of the Eleventh International World Wide Web Conference, Honolulu, HI, vol. 45 2002
20. Prakash, C.J., Rohini, P.M., Ganesh, R.B., Maheswari, V.: Hybrid reliability model to enhance the efficiency of composite web services. In: 2013 International Conference on Emerging Trends in Computing, Communication and Nanotechnology (ICE-CCN), pp. 79–83. IEEE (2013)
21. Sargolzaei, M., Santini, F., Arbab, F., Afsarmanesh, H.: A tool for behaviour-based discovery of approximately matching web services. In: Hierons, R.M., Merayo, M.G., Bravetti, M. (eds.) SEFM 2013. LNCS, vol. 8137, pp. 152–166. Springer, Heidelberg (2013). doi: 10.1007/978-3-642-40561-7_11
22. Shadi, M., Afsarmanesh, H., Dastani, M.: Agent behaviour monitoring in virtual organizations. In: 2013 IEEE 22nd International Workshop on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE), pp. 9–14. IEEE (2013)
23. Tabatabaei, S.G.H., Kadir, W.M.N.W., Ibrahim, S., Dastjerdi, A.V., et al.: Integrating discovery and composition of semantic web services based on description logic. J. Comput. Sci. Inform. Electr. Eng. **3**(1) (2009)
24. Terlouw, L.I., Albani, A.: An enterprise ontology-based approach to service specification. IEEE Trans. Serv. Comput. **6**(1), 89–101 (2013)
25. Hao, Y., Zhang, Y., Cao, J.: WSXplorer: searching for desired web services. In: Krogstie, J., Opdahl, A., Sindre, G. (eds.) CAiSE 2007. LNCS, vol. 4495, pp. 173–187. Springer, Heidelberg (2007). doi:10.1007/978-3-540-72988-4_13
26. Yu, T., Lin, K.-J.: Service selection algorithms for composing complex services with multiple QoS constraints. In: Benatallah, B., Casati, F., Traverso, P. (eds.) ICSOC 2005. LNCS, vol. 3826, pp. 130–143. Springer, Heidelberg (2005). doi:10.1007/11596141_11