Supporting Software Services Discovery and Sharing in Collaborative Networks

Alexandre Perin-Souza, Ricardo J. Rabelo
Department of Automation and Systems - Federal University of Santa Catarina – Brazil
{perin, rabelo}@das.ufsc.br

Abstract. Collaborative Networks (CN) realization fundamentally relies on the need of *collaboration* among involved partners. CN members who have SOA-based solutions keep the involved web-services at their local silos. This means that the CN potential in terms collaboration could be enlarged and reinforced if such silos could be opened up and shared among CN members, so decreasing development and hosting costs. This is particularly relevant as CN members are mostly composed of small companies. This paper presents a result of an exploratory research, proposing a model for dynamic service discovery based on Software-as-a-Service business model, which considers not only services' functional requirements, but also the requested QoS and business processes' context. A prototype has been developed to show the concepts. A preliminary assessment from the ICT and CN perspectives is given in the end.

Keywords: BPM; SOA; Web Service Discovery; QoS; Business Process.

1 Introduction

Market dynamics has been imposing many challenges to organizations nowadays in order to adapt themselves and hence to be agile. Agility should be tackled under several perspectives. In this context, the adoption of the Collaborative Networks (CN) paradigm by organizations has been considered one of the most promising strategies. Its ultimate goal is to enable networked organizations to agilely define and set up relations with other organizations as well as to be adaptive according to the business environment conditions and current organizations' autonomy levels [1]. This requires, however, more effectiveness, flexibility and collaboration in businesses.

From the ICT perspective, such requirements have demanded more advanced infrastructures [2]. A number of prominent ICT approaches have been proposed in this sense. The most impacting ones are SOA (Service Oriented Architecture) and Utility paradigms [3]. SOA can be generally defined as an architectural paradigm for components of a system and interactions between them, where its functions are viewed as software modules called *services* [4]. *Utility* associates the idea that a service should be discoverable and usable anytime, anywhere, like electricity.

The problem is that, in practice, CN members who have SOA-based solutions keep the involved services (typically implemented using web-services technology) in their local silos. This means that the SOA potential in terms of reuse could be extended and highly increased if such silos could be shared among CN members, i.e. if all services could be accessed by any member so enlarging and reinforcing

collaboration while development and hosting costs are decreased. This seems particularly relevant as CN members are mostly composed of MSMEs[1], without many conditions to maintain IT infrastructures and costly staff. In this sense, CN members could be both clients and service providers [2].

This is the underlying motivation of this paper. CN members - VBE, VE, VO, PVC[2] members and other type of companies (e.g. logistics operators and software providers) have the potential to enlarge their collaboration via an interoperable and transparent collaboration "cloud". Several works have been developed having this future CN scenario in mind, such as virtual machining [5], knowledge search and sharing over CN's information repositories [6], virtual shop-floor [7] and CN ICT infrastructures [2]. This paper intends to contribute within this wider collaborative scenario seeing services under the Utility paradigm point of view, proposing a supporting model for software services discovery when partners share their services.

Supporting this scenario requires coping with many issues, of different levels of complexity. One of the issues refers to the access mode and business models that can be aligned to this collaborative scenario [8]. In the SOA context, SaaS (*Software-as-a-Service*) [9] has arisen as one of the most powerful models. Using SaaS, clients (i.e. CN members) can flexibly build and adapt their services portfolio according to their needs [10] instead of getting held to single vendors of monolithic software packages whose full set of functionalities are often few used/accessed. With SaaS, services are accessed remotely, upon request, paid-per-use, based on contractual rules specified in SLAs (*Service Level Agreement*) [11] for hosting, managing, providing access to them following QoS (*Quality of Service*) levels, no matter where the services providers are and how services have been deployed [12]. This seems suitable to CN members due to their intrinsic independence, autonomy, large geographic distribution and heterogeneity. A second issue refers to the discovery of services that are shareable, and this is what this paper is more about.

In this services discovery scenario, client and provider perspectives should be taken into account. From the CN clients' side this means how expressing the desired service as well as how finding, selecting and binding services to their composite SOA-based applications. From the CN providers' side, this means how publishing and making their services available. Besides that, CN clients must feel confident to access not only a given functionally-compliant service, but ideally to the most suitable service regarding the current computing environment (in terms of required QoS) in use and business process' context.

Therefore, this paper presents an ongoing work, corresponding to an exploratory, applied and essentially qualitative research, which aims at contributing to answer the following research question: how to support those two mentioned issues considering CN characteristics in way to leverage web services discovery and sharing ? In this sense, a strongly ICT standards-based model that supports dynamic services discovery over largely distributed CN services providers is proposed. It considers SaaS architectural model, dealing with not only services' functional requirements, but also the requested QoS and business processes' context.

---

[1] MSME – Micro, Small and Medium Enterprises.

[2] VBE (Virtual Organization Breeding Environment), VE (Virtual Enterprises), VO (Virtual Organization), PVC (Professional Virtual Communities).

The paper is organized as follows. Section 1has framed the problem and the tackled scenario of services discovery and sharing among CN members. Section 2 summarizes the main concepts and literature overview on services discovery. Section 3 presents the proposed model and section 4 presents a software prototype. Section 5 discusses about the preliminary results. Some conclusions are presented in section 6.

2 Basic literature overview

The problem of web services discovery involves many complex issues, such as: i) technological heterogeneity and low level of interoperability; ii) ambiguity of concepts due to differences of domain applications; and iii) limitations in the technologies used to design services [13]. There are plenty of works and visions on services discovery in the literature. The review here presented encompasses only the considered most relevant works for the purpose of this paper.

Two basic strategies resume the works on service discovery presented in the literature: static and dynamic discovery. There are a number of advantages and disadvantages in both. In the static way services are discovered at design time and they are immediately associated / bind to the given SOA application. However, the previously chosen service can be not available when the service is invoked. The main advantage of the dynamic approach is the possibility of replacing services by others or of choosing the most appropriate one, on-the-fly. However, this strategy usually presents higher discovery time (as all discovery actions should be done at execution time) and demands more sophisticated levels of intelligence from the discovery algorithm. This intelligence can be in the form of criteria relaxation [14] or other techniques such as classification systems, providers' reputation, costs negotiation and quality of service development process [15].

Existing models and approaches for services discovery can be grouped into four dimensions: *information retrieval, architecture, QoS* and *standards* [11]. In general terms, in the information retrieval dimension the focus has been put on semantics in way to provide greater precision in the services selection [14] [16]. In the architectural dimension, works have focused on aspects such scalability, security, availability, etc. [17]. In the QoS dimension there are three basic fields of research [18] [19]. The first one refers to the definition of attributes and metrics for QoS. The second one involves the establishment of more comprehensive and robust frameworks to represent, select, verify and maintain QoS attributes. The third field involves the development of *ad hoc* strategies to solve specific cases involving QoS. In the standards dimension, the initiatives have focused on interoperability (in heterogeneous environments). ICT standards like UDDI [20], SOAP [21], WSDL [22] have been intensively used.

The problem is that the envisaged collaborative scenario requires a more holistic and comprehensive view about services discovery. Besides requiring a joint view of those four dimensions, other dimensions not tackled in any of the reviewed works are also necessary, namely the consideration of the business level. In spite of the very high complexity all this represents and challenges that have to be faced, this proposed model intends to contribute to this.

3 Proposed Model

3.1 Essential Rationale

The essential proposal's rationale is fundamentally related to a tighter integration between BPM and SOA as a way to enhance services discovery quality. Actually, the combination of BPM (Business Process Management) [23,24] and SOA [25] has been considered as the current state-of-the-art approach to improve the quality of integration and alignment of business and IT [26].

Very roughly, it can be said that BPM allows managers to express business processes' rules whereas SOA acts as the link with the IT level. In spite of the availability of several BPM and SOA software solutions in the market, the complete integration of them is a tremendous challenge in practice regarding the required organization and IT changes for a company. One issue of this challenge refers to the problem of web services composition, whose ultimate goal is to bind services (i.e. to compose the SOA-based application) for the involved business processes (BP).

State of practice on BPM&SOA integration shows that companies use to make the composition in a fixed, not much assisted and hard-coded way, i.e. once managers specify the processes (at BPM level) the IT engineers bind them with existing (or to be developed) web services. The execution is further carried out via e.g. BPEL (Business Process Execution Language) tools [27]. Moreover, most managers use to make that specification without considering already available business process models standards, such as UBL (Universal Business Language) [28] or RosettaNet [29], which has the potential to mitigate semantics interoperability problems when integrating (SOA) applications. However, even when this is considered, its usage gets normally restricted to the process modeling moment (business level), and not extended and passed to the SOA layer (IT level). This means, for instance, that BPs' context is lost for further services discovery purposes. If on one hand this conventional way somehow works in the companies, on the other hand it prevents them from being more flexible and agile in their business. Each new process or modification requires reprogramming and rebinding, which is usually costly and time consuming, especially for CN members, whose core business does not use to be web services development. Even so, companies are responsible for managing this asset as well as for handling the involved services' QoS.

Another problem refers to this perspective of allowing largely distributed and heterogeneous CN members (including small software providers) to share their services under the SaaS (Software-as-a-Service) [9]. This scenario can be particularly interesting for CN members as one of the ways they can somehow outsource their TI needs. Besides that, this scenario provides the potential for choosing (even on-the-fly) the most adequate service for the current BP's requirements (from the functional and non-functional points of view, including different web services' costs) and computing environment a given CN manager is in.

3.2 Issues to tackle

Coping with all the requirements involved in the envisaged collaborative scenario of services discovery and sharing is extremely complex and also involves several challenges at business and IT levels [30].

In the proposed model for services discovery within a CN scenario, there are five basic issues to be considered: *i) what to express; ii) how to express the desired service; iii) who expresses the desired service; iv) who evaluates and how discovery results are evaluated; and v) when expressing and searching.*

The first point refers to which information should be expressed to specify the desired service for the given BP. In the proposed model, this involves: i) functional aspects (service's name and operations); ii) inputs and outputs; iii) expected QoS values; and iv) BP's context.

The second point refers to services expression including the components and the relationship between them. Natural language, formal or dedicated languages are examples of how to do this. The proposed model adopts ontology. They create a uniform vocabulary to be used by the discovery mechanisms, besides improving discovery process accuracy and precision.

The third point refers to who informs the desired service. Traditionally, the SOA application designer is the one who explicitly specifies the details of the desired service. Considering the role of BPM systems in this work, four ways might be considered: Automatic (A), Strongly Based on Designer (SBD), Semi-Automatic (SA) and Assisted (As). In the Automatic way (A), the discovery task is left to the BPM environment. It automatically identifies needs, makes changes in the expression and determines which service will be selected and be bind to a given BP. This process occurs without designer intervention. In the SBD way is somehow the opposite case, where the designer is the responsible for indicating the required service and for evaluating and deciding about the most adequate service to be selected. The designer interacts with the BPM environment intensively. In the SA way is a hybrid way. The BPM does the whole process but asks to the designer for some constraint relaxation in the case no expected service is found out. In the Assisted (As) way, there is a closer interaction between BPM environment and designer. The designer specifies service's requirements whereas BPM environment informs the BP's context. The evaluation of the discovery result is jointly performed.

The fourth point involves determining who evaluates the discovery results and how this evaluation occurs (assisted, without designer participation, etc). Four situations are considered: i) more than one 'perfect' service (i.e. there is full compliance between desire and outcome), which means having the need for ranking and selecting the most appropriate service; ii) just one perfect service; iii) none service; iv) one partial service (any 'perfect' service); and v) more than one partial services.

The fifth point refers to at which moment the activities of identifying the BP, its requirements, etc. as well as the results evaluation will be carried out by of the discovery actions. This point is detailed in the next section.

3.3 Operation

A key aspect in the proposed model is a change in the way the problem of dynamic services discovery have traditionally been seen. In this work, the problem is split into two phases: design phase and running phase (Figure 1). The essential rationale of this separation is trying to prevent CN managers from handling IT issues as much as possible. In this sense, at BPM level, it is considered that they are very familiarized with business rules and can specify how business processes should be executed. On the other hand, at the running time, they shouldn´t get too much worried about the services discovery and this should be essentially left to the SOA environment / discovery system to do it.

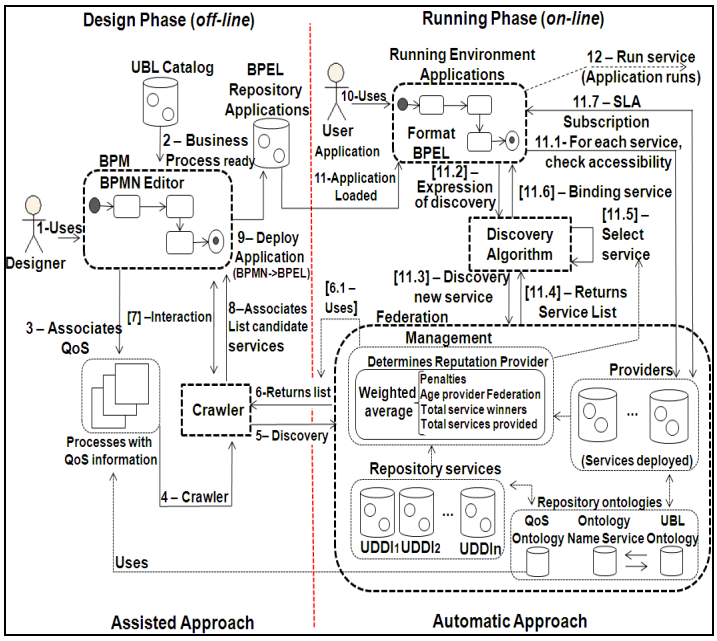

Fig. 1. Overview of the Model operation.

The first phase basically refers to the specification of the SOA application to be built, including capturing the given BP's context and indication of the required QoS, which are crucial elements for a more proper discovery (left side of Figure 1). This is done off-line and CN managers are assisted by the composition environment (see section 3.2 - third point). Initially, the (s)he uses a BPM environment for composing applications (step 1) - also showed in Figure 2 - which offers the access to a standard *BPs catalog* (step 2). This catalog, an additional element of the model, comprises all the business processes of UBL standard (the one chosen in this exploratory work), meaning that all composite applications will be compliant with a standard or with a specialization of it. Yet, it is possible to identify and to capture the BP's context. This speeds up the application design and provides greater semantic richness to the discovery since services (from the several CN providers) are published taking a UBL standard ontology into account.

In step 3, CN manager together with an IT technician assign QoS attributes for each BP's activity. A QoS ontology (Figure 1 right / bottom) is used for this, and it is also used by the CN providers when publishing their services. The search expression is then assembled. The expression is structured as a 3-tuple: the wanted web service W = (*F,O,Q*), where F represents the set of functional requirements (i.e. the required functionalities), O represents the BP's ontology, and Q the non-functional requirements (i.e. QoS). As an example, let´s take the UBL general process related to *purchasing*. According to the UBL ontology, *ordering* is one of its subprocesses. *Ordering* has some I/O to be respected in terms of number of arguments and types. The other sublist refers to QoS, where some values are assigned to the several attributes comprised in the QoS ontology (e.g. *response time*).

An internal crawler service is activated (step 4) having the search expression as input. The crawler begins searching (step 5) for services in the CN members' services repositories with the aim of bringing a list of possible services that matches the subprocess requirements. It is also important to mention that the crawler acts in "background", i.e. that list is retuned at design time, and not at the running time. The underlying strategy is to decrease the whole discovery time, avoiding an exhaustive search when the composite SOA application would be in execution.

A list of services is returned (step 6). This list is ranked according to the CN providers' reputation (step 6.1 - historical, penalties, etc). As mentioned in the previous section, a 'perfect' matching may be not occurs, and the crawler should handle this (step 7). If the list contains just one service, it is already settled to the subprocess. If the list is empty, the crawler asks for a requirements relaxation for a new search, or leaves the option (or risk) of keeping the same requirements and doing the search at running time. If the list has more than one service, they are stored as XML files for further treatment at execution time (see below). In step 8, still at the design phase, this list of candidate services is passed to the BPM environment. The SOA application is then composed, it is converted to BPEL [27] format and gets available in stand-by to run (step 9).

It is important to clarify one aspect related to the used ontologies and the matching. Actually, the use of ontologies is a strategy to face semantics interoperability problems when consumers go for a discovery and providers publish their services. UBL and QoS ontologies (previously populated) act as a common / agreed set of terminologies and concepts used to express the BPs and QoS. This makes possible that the matching in the discovery can be "syntactic", via a comparison of the same (semantic) terms used in the adopted ontologies.

The second phase of the model – the on-line phase – starts when the CN manager or collaborator wants to execute a given application (that one previously composed). At this time, it is relatively transparent to him that it is a SOA-based application. He navigates through the process / BPEL repository and chooses which application he wants to run (steps 10 and 11, Figure 1 right side). A runtime task checks if the pre-selected web services are actually available (step 12), invoking the selected ones in the case they are ok. Otherwise the second one from the list will be picked up, and so forth. In the case none of them are available then the whole discovery activity is triggered again, with a new possible list, etc. Once everything is settled for a given service, the respective SLA (Service Level Agreement) is dynamically generated (step 11.7) and integrated into the BPM environment, as proposed in [31]. Thus, and helped

by a BPEL engine, the desired composite SOA application (e.g. the *purchasing* application in this case) is ready to run, with the most appropriate services (step 12).

This proposed model has a set of assumptions. Two of them are important to be highlighted. The first assumption is what we call 1:1 relation. This means that the composition model is prepared to find and to compose 1 service for each 1 BP's subprocess activity. Taking the same example used along the paper and according to UBL ontology, *purchasing* process is composed of a set of subprocesses e.g. *ordering*. Therefore, the model will understand that a discovery action (considering functional, QoS and context requirements) will be carried out aiming at finding a sort of possible web services for *ordering* but only one (1) web service will execute it. In other words, this assumes that services providers would have implemented one (1) service per subprocess (eventually they can have different versions, associated or not to different business models, depending on the client). The second assumption is that the so-called CN manager and IT technician are expected to be experienced persons, who understand company's processes and rules (and UBL ontology) as well as have the 'feeling' about suitable QoS metrics (and ontology) for the SOA application that is going to be built up.

4 Implementation and Preliminary Assessment

In order to evaluate and to compare the value added of the proposed model against the so called traditional model (i.e. without consideration of QoS and contexts), two prototypes have been implemented.

About the UBL ontology, it was based on the UBL standard specification and was expressed in BPMN (Business Process Modeling Notation). The catalog was built on top of this, as a plugin of IBM WebSphere BPM suite. About the QoS ontology, it was conceived based on a high-level ontology [32] and QoS definitions from two other works [33, 34]. Both ontologies were modeled using Protégé tool. It is assume that CN members / VBE share such ontologies.

Web services repositories were implemented based on the UDDI 3.0 specifications, using *bindingTemplate* elements [20] to save QoS aspects.

Figure 2a shows one of the several interfaces of the proposed model's environment. It is the main one through which a CN client indicates which service is requested (using the UBL ontology; in this case, the *ordering* service) as well as the expected QoS values (using QoS ontology). Figure 2b shows a result of the discovery. For each desired service, the system presents the list of 'perfect matching' services (see section 3.3 / step 6) with their name ("ordering", which is related to the UBL *purchasing* process' context), the CN providers' identification and the QoS values they support.

A hypothetical scenario was created in order to preliminary test the model. It assumes the existence of 10 CN providers offering 100 web services in total. All registries and providers were deployed and placed in a local network. QoS values were randomly assigned to them. These services are related to the *purchasing* UBL process, which in turn has four subprocesses: *Ordering, Sourcing, Payment and Billing*. 25 similar (UBL compliant) services were equally developed for each of these sub-

processes and randomly deployed over the providers. UBL and QoS ontologies were used for services publishing.

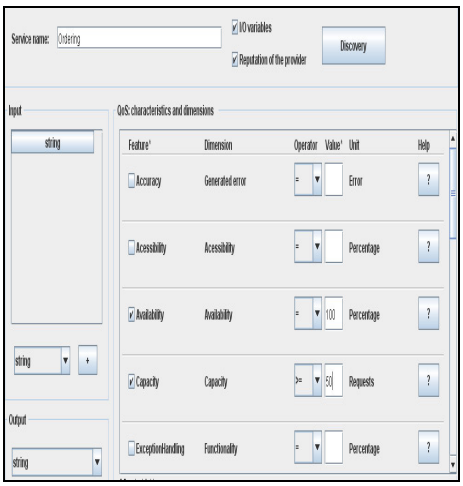

Fig. 2a. Interface for service discovery

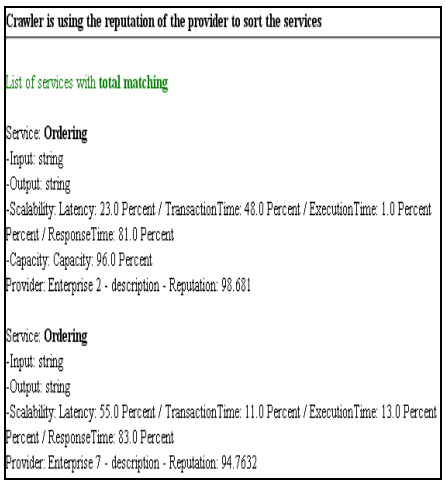

Fig. 2b. Example of discovery results

Three search queries were idealized to try to make a preliminary evaluation of the model's approach (Figure 3). The first query was composed of four discoveries, one for each of those four subprocesses. This query did not take into account QoS information, i.e. it only dealt with functional requirements, which corresponds more or less to the traditional scenario of discovery. After a search 100 services (25 per subprocess) have returned as all registered services were semantically compliant with those subprocesses terminology and functional requirements. In the second query QoS information was added to the search expression. The number of returned services (8) was evidently lower than in the first query. This represents the result of the QoS filtering, meaning that only 8 services (out of 25) actually had 'perfect matching' and hence are potentially suitable for the given process. A complementary filtering that can be used is the consideration of the service / providers reputation.. The third query tried to explore a more "realistic" scenario, which considers not only 'perfect' services but also services with partial (QoS and/or functional requirements) matching. As a result of the search the returned list of services (12) was numerically greater than the second query. Now the designer's job would be more difficult. If in the second case his job would be selecting the best service based on the reputation ranking (as all services had matched everything), in this third case a more careful analysis should be done. This means that the designer should go through each of possibility to see the eventual need for requirements relaxation (see section 3 step 7).

In terms of global performance, the discovery process itself (i.e. not considering the design time where the designer manipulates the catalog, assigns QoS, etc.) was executed in seconds. However, this is evidently biased to the fact that all UDDIs and services were deployed in a local network. In a real scenario of largely distributed CN providers and members in general, network latency (besides reliability) is a must to be considered. In terms of computational efficiency, the current version of dynamic discovery and the crawler algorithms have an efficiency of linear time $O(n)$.

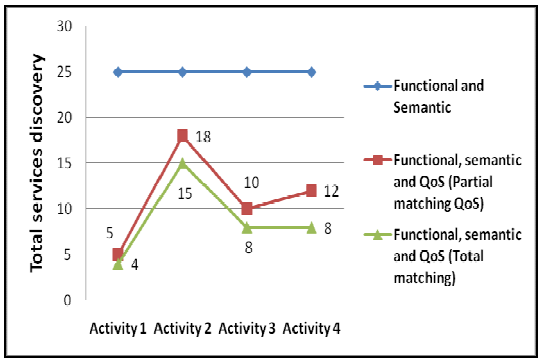

Fig. 3. Result of the queries 1, 2 and 3.

5 Conclusions

This paper has presented an ongoing and exploratory work on dynamic services discovery. It copes with some characteristics of CN as well as with current trends in integration of business and IT levels, namely considering BPM and SOA paradigms.

The proposed model is strongly grounded on standards, at all tackled levels. It facilitates interoperation and mitigates problems for a larger acceptance of more advanced concepts – BPM and SOA in the case – by CN members.

Compared to the state-of-the-art, this model adds value when brings BPM closer to IT/SOA level, applying standards, allowing more adequate and dynamic services selection when considering BP's context and QoS, and providing an advanced and sustainable business model (SaaS) for services sharing among CN members.

The model is currently instantiated with UBL open standard, although it is prepared to deal with other equivalent ones, such as RosettaNet or some future initiative that can formalize CN-related business processes. It is foreseen that plenty of small software companies and CN members over the world will be engaged in offering and using standard-compliant services under the SaaS model in the near future. A model like the proposed one can be a good starting point for tacking this so challenging scenario.

From the exploratory perspective of this research, it can be said that the proposed model has the potential to solve the "theoretical" part of the problem (based on the chosen approach), which refers to how CN members can share their web services and hence can enlarge their collaboration and reinforce trust building. This can give suitable conditions for a more agile adaptation in the members' business processes, in particular when they need to cope with specific business opportunities forming a virtual organization.

However, from the applied perspective of this research, a model like that cannot be seen as the solution for all issues involved in services discovery and services sharing among CN members. For example, SaaS in an emergent area and there is not solid theoretical foundations about it yet. Trust, adequate ICT infrastructures, security,

cultural changes, BP (re)organization, among many other aspects, are examples of additional and complex issues that should be dealt with. In spite of this, it is believed that this model has a structure that can be used as a starting and useful instrument to allow a wider collaboration among CN members, preserving their independence, autonomy and heterogeneity.

The problem related to CN members services' quality and trustworthiness can be talked by means of applying reference guidelines for SaaS development, such as proposed in [35].

On the other hand, the own VBE (which involved CN members would come from) can adopt an institutional position stimulating CN members to share their services. In the future, companies can even make some profit out of this since the access to their services (considering that they would have to maintain them anyway) would be on-demand and paid-per-use. Yet, strategic benefits can be a good argument for its adoption as CN members can become more agile and feel better prepared to enter in a new VO as new required software services may be, in fact, available, preventing them from new services development or acquisition, in opposite to the case without any sharing.

References

1. Camarinha-Matos, L. M. and Afsarmanesh, H. Collaborative Networked Organizations: a Research Agenda for Emerging Business Models. Kluwer Academic Publishers (2004).
2. Rabelo, R. J. Advanced Collaborative Business ICT Infrastructures. In: Methods and Tools for Collaborative Networked Organizations, Springer, pp.337-370 (2008).
3. NESSI Strategic Research Agenda - Framing the future of the Service Oriented Economy. Version 2006-2-13 (http://www.nessi-europe.com/documents/NESSI_SRA_VOL_1_20060213.pdf) ICT for Enterprise Networking (cordis.europa.eu/ist/directorate_d/en_intro.htm). (2006)
4. Singh, M.; Huhns, M.; Service Oriented Computing, Wiley, (2005).
5. de_Souza_Jr., J.L.N.J., et al., An Internet-oriented Management and Control System in a Distributed Manufacturing Environment. International Journal of Manufacturing Research, 2009. 5: p. 5-25. International Journal of Manufacturing Research. (2009).
6. Tramontin_Junior, R.J., R.J. Rabelo, and H. Chirab, Customizing Knowledge Search in CNOs through Context-based Query Expansion. Production Planning & Control, 2010. 21(2): p. 229-246. Production Planning & Control. (2010).
7. Ribeiro, L., J. Barata, and A. Colombo, Supporting agile supply chains using a service-oriented shop floor. Engineering Application Artificial Intelligence (2009).
8. Borst, I., *et al*., Technical Report (Deliverable) D62.2 ICT-I Business Models, in http://www.ve-forum.org/default.asp?P=284.
9. Software & Information Industry Association (SIIA). Software as a Service: Strategic Backgrounder. http://www.siia.net/estore/ssb-01.pdf. (2001).
10. Tsai, W.T. Service-oriented system engineering: a new paradigm. In: Service-Oriented System Engineering, SOSE 2005. IEEE International Workshop. 2005: p. 3-6 (2005).
11. IBM-WSLA. Web Service Level Agreement (WSLA) Language Specification. V 1.0. http://www.research.ibm.com/wsla/WSLASpecV1-20030128.pdf. (2003).
12. Ma, D. The Business Model of Software-As-A-Service. In: Services Computing, 2007. SCC 2007. IEEE International Conference. p. 701-706 (2007).

13. Garofalakis, John et al. Contemporary Web Service Discovery Mechanisms. Journal of Web Engineering, Rinton Press,v. 5, n 3, p. 265-290, (2006).
14. Kokash, Natalia. Web Service Discovery With Implicit QoS Filtering. Proceedings of the IBM PhD Student Symposium, (2005).
15. Mello, E. R. ; Fraga, J. S. ; Wangham, M. S. Using a trust model for the composition of Web Services [in Portuguese]. In: Brazilian Symposium on Computer Networks (2009).
16. Kourtesis, Dimitrios *et al*. Web Service Discovery in a Semantically Extended UDDI Registry: The Case of Fusion. Book Series IFIP International Federation for Information Processing. Book Establishing The Foundation Of Collaborative Networks. Publisher Springer Boston, v. 243, p.547-554, (2007).
17. Lamparter, Steffen; Anklekar; Anupriya; Studer, Rudi; Grimm, Stephan. Preference-based Selection of Highly Configurable Web Services. Alberta, Canada (2007).
18. Ran, Shuping. A Model for Web Services Discovery With QoS. Journal SIGecom Exch. V. 1, n. 1, (2003).
19. Wang, X. et al. A QoS-aware Selection Model for Semantic Web Services. In Service-Oriented Computing – ICSOC. (2006).
20. OASIS. UDDI Specific. V 3.0.2. (2004).
21. W3C. SOAP Specification. V. 1.2. (2007).
22. W3C. Web Services Description Language Specification. V 2.0. (2007).
23. ORACLE. State of the Business Process Management Market. An Oracle White Paper. www.oracle.com/technologies/bpm/docs/state-of-bpm-market-whitepaper.pdf (2008).
24. BPMINSTITUTE. State of Business Process Management. www.bpminstitute.org (2006).
25. W3C. Web Services Architecture. www.w3.org/TR/2004/NOTE-ws-arch-20040211(2004)
26. Kamoun, Faouzi. A Roadmap Towards the Convergence of Business Process Management and Service Oriented Architecture. Ubiquity Journal, v. 8, n. 14, p. 1-8. (2007).
27. OASIS. Web Service Business Process Execution Language V 2.0. http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html (2007).
28. OASIS. Universal Business Language V2.0. http://docs.oasis-open.org/ubl/os-UBL-2.0/UBL-2.0.html (2006).
29. ROSETTANET. Business Process Specific. (PIP). (2008).
30. Dorn, J.; Grün, C.; Werthner, H. & Zapletal, M. From business to software: a B2B survey. Information Systems and E-Business Management, Springer, v.7, 123-142 (2009).
31. Cancian, Maiara Heil; Rabelo, R. J. ; Wangenheim, C. G. V. An approach for the generation of SLAs for Software-as-a-Service [in Portuguese], In: 8th I2TS Int. Conference. Florianopolis. Brasil (2009).
32. Tondello, G. F. Sematic Specification of QoS: the QoS-MO Ontology [in Portuguese]. M.Sc. Thesis, Federal University of Santa Catarina (2008).
33. OMG (Object Management Group). UML Profile for Modeling QoS and FT Characteristics and Mechan. Specification, v1.0. (2006).
34. W3C. QoS for Web Services: Requirements and Possible Approaches. http://www.w3c.or.kr/kr-office/TR/2003/ws-qos/ (2003).
35. Cancian, M. H.; Rabelo, R. J., Wangenhein, C.G.. Supporting Software Services' Trustworthiness in Collaborative Networks, to be presented in 11th IFIP Working Conference on Virtual Enterprises, Saint Etienne, 2010.