For a Dynamic Web Services Discovery Model for Open Ecosystems of Software Providers

Ricardo J. Rabelo, Hernesto A. Ruiz, Maiara H. Cancian

For a Dynamic Web Services Discovery Model for Open Ecosystems of Software Providers

Ricardo J. Rabelo[1], Hernesto A. Ruiz[1] and Maiara H. Cancian[2]

[1] UFSC - Federal University of Santa Catarina
Florianopolis (SC) Brazil
ricardo.rabelo@ufsc.br ; hernesto4706@gmail.com

[2] Estácio University Florianopolis
Rodovia SC401 km 01 – Florianopolis (SC) Brazil
maiara.cancian@estacio.br

Abstract. With the increasing adoption of business models relaying on Cloud Computing, digital platforms and servitization, IT providers have been transforming their solutions into services-based, having as a goal facilitating their binding to clients' applications. In this formed wide and open service-oriented environment, classical static services binding approaches are limited to support the business dynamics of Virtual Organizations (VO). As services would assure VO members' systems operation, a bad services selection may hazard that. We argue, though, that services selection criteria, largely based on QoS and costs, are not enough to provide higher confidence to VO members when selecting services from such open ecosystem of service providers. In this line, this paper proposes a more comprehensive and multi-criteria decision-making model for dynamic services discovery. A prototype has been implemented and results are discussed.

Keywords: Service Oriented Architecture, Dynamic Services Discovery, Web Services, Ecosystem, Industry 4.0, Collaborative Networks, Virtual Organization

1 Introduction

Collaborative Networks (CN) play a key role in Industry 4.0 realization. It provides theoretical foundations for some of its key issues, as networking, vertical and horizontal integration, sharing, co-engineering and interconnectivity, immersed in business models grounded on the Internet of Things, Services and People [1].

This has been imposing a much higher flexibility from companies' systems architectures. Firstly, in their classical transaction-based systems, usually designed in a quite fixed, pre-defined and hierarchical/vertical way based on the five-layer ISA-95 model[1]. Secondly, in their CN supporting systems, designed to execute *ad-hoc*, non-hierarchical/horizontal, interlayered and inter-company business processes (BP) [2,3].

Computing reality has changed significantly in the last two decades. Cloud computing, service orientation and digital platforms, besides newer business models built up on top of this, have made disparate IT providers from digital ecosystems to start offering computing infrastructures and applications *as-a-service*, and that can be accessed remotely and on-demand [2,4]. Following this trend, many industries and SMEs have changed their IT strategy. From the *acquisition* of large, off-the-shelf, on-

[1] https://www.isa.org/isa95/

premises and sometimes very expensive software packages and lock-in licenses, they have been gradually shifting to *servitization*, accessing only specific and much smaller software's functionalities when needed and paid-per-use [5]. These functionalities are called and modeled as *services* and generally called as *web services*.

This shift also happens because: i) companies have realized that some of the required functionalities for their BPs/systems are not feasible or interesting to be developed and/or maintained by their own as they may be already available in the market/ecosystem; ii) companies' business models, their products, management methods, etc., are more often changing, meaning some unpredictability in their near future software needs; and iii) only large IT providers (as *Google*, *Amazon*, etc.) are actually able to develop and maintain some complex software as IT and technology rapidly evolve, offering them as-a-service via their APIs [6,7].

Within this context and regarding the usual SME limitations at several levels, this paper exploits the vision where companies adopt that service-oriented and sharing strategy when they form a Virtual Organization (VO) [1,2], and that a services-based architecture/system would be dynamically created to support the execution of the involved (intra- and inter-organizational) VO's BPs [2].

In this vision, all such IT services providers would belong to diverse digital ecosystems, similarly to a wider VBE[2]. They would publish and make their IT assets available (their software services and, indirectly, their required computing infrastructure) in a logical entity called as *services federation* [2,4,8]. Providers keep their autonomy and legal independence, owning and providing software services' implementations and descriptions as well as the respective technical and business support. Companies can have access to them *as-a-service* [4,8].

In this scenario, companies could then have access to a "infinite cloud" of services (the federation). Assuming there would potentially (and naturally) exist several software services functionally equivalent one to another, and that an inadequate selection may hazard the VO success and efficiency, this paper addresses the problem of how to select the most suitable services for the very current VO's BP needs.

This problem is known as *dynamic services discovery* [9]. In general, the approaches for services selection are essentially based on evaluating some services' quality of service (QoS) attributes [9]. We argue that the mentioned ecosystem and VO scenarios require additional aspects to consider so that companies can be more confident when selecting services for their daily operations. Yet, that companies could have the possibility to weight such aspects considering their current priorities and IT policies.

In this direction, this paper proposes a more comprehensive set of decision criteria for multi-criteria services selection, which works within a wider discovery model.

This is an ongoing proof-of-concept work being developed under the *design science research* (DSR) methodology, looking at a SME cluster of 34 Industry-4.0 IT software providers placed in the South of Brazil[3] as an initial basis for the criteria elicitation.

---

[2] VBE - Virtual organization Breeding Environment [1].
[3] https://manufatura.acate.com.br/

2 Basic Concepts & Summary of Literature Review

2.1 Dynamic Services Discovery

The selection of a software by companies normally takes a long time, is manually done, and use to consider aspects as price, supported functionalities, documentation, technical assistance, used technology, lock-in risks, providers branding, historical partnerships, impact on legacy systems and integration costs, among others [6,10]. Many of these aspects are somehow embraced by the 31 *characteristics* of ISO/IEC 25010 standard[4] for systems and software quality evaluation. Yet, once the selection is settled, contracts usually last at least a month for SaaS mode, and much longer for on-premise software.

Regarding the beforehand mentioned vision and the intrinsic dynamics of VOs, such long selection process and the static/fixed binding of software services to BPs prevents VO formation and evolution from both being more agile and benefit from the huge variety of services nowadays ecosystems can offer [2,9].

Service discovery refers to locating web services, evaluating their suitability considering functional and non-functional requirements, and to finally selecting one. It is considered as dynamic discovery when it is performed at run time [9,11].

Searching for the suitable web services out of all the available ones in large-scale distributed federation-like entities can be very computationally complex [12]. Besides the selection criteria to be defined, it should also try to: i) respect the individual requirements and technology restrictions of every single BP's activity against the ones supported by every single candidate service; and ii) respect the end-to-end QoS of the entire BP all those activities are related to [13].

2.2 Literature Review

The envisaged discovery model involves several issues, for which many works in the literature have proposed different approaches. Given space restrictions, this section presents just a highlight of some of them.

Much research has been done about dynamic services discovery over the last fifteen years. Obidallah *et al.* (2016) [14] present a large overview of the different approaches for services discovery. In terms of selection criteria, QoS-based are by far the most adopted ones. However, no mentions were made to more appropriate criteria for discovery in large scale services repositories from disparate ecosystems of IT providers.

Camarinha-Matos *et al.* (2011) [15] created a matching-maker environment in a cloud where different partners of CN can offer their (non-software) services respecting the technical demands from a given manufacturer. Osorio *et al.* (2011) [16] and their subsequent works have presented an evolving services-based platform to support payment actions by users of tolling systems involving ecosystems of disparate providers. Thanks to open reference specifications, they can plug and play their solutions in the platform. However, these two works do not support dynamic discovery.

[4] https://iso25000.com/index.php/en/iso-25000-standards/iso-25010, former ISO/IEC 9126.

Correa *et al.* (2014) [17] devised a AHP-based multicriteria framework to select the best logistic service providers for given businesses out of a logistic providers cluster based on their historical performance regarding QoS indicators. Bezerra *et al.* (2018) [12] developed a system to dynamically discover web services from a services federation when faults are detected in SOA compositions. Perin *et al.* [9] implemented a web services dynamic discovery model integrated to the BPM level in way that SOA compositions can be created and dynamically evolve as businesses change. Bandara *et al.* (2015) [18] implemented a prototype where users can balance criteria regarding current processes' needs. These four works are just examples to show that QoS-based criteria are the most adopted approach in services discovery, sometimes combined with multi-criteria methods. Cancian *et al.* (2015) [19] conceived a model with 22 processes that should be supported to manage the federation/ecosystem of SaaS providers.

Despite the importance of the many contributions of current works, none works have proposed more comprehensive selection criteria better suited for dynamic services discovery in open and evolving IT SaaS/services providers ecosystems.

3 Services Selection Criteria

3.1 Elicited Criteria

Following the DSR methodology, an initial set of selection criteria for services discovery was created after a generalization and terminology harmonization based on a literature review. It was further submitted and discussed in a meeting with the mentioned IT cluster. Given that this research proposal can be considered as a state-of-the-art scenario, the discussions were open and tried to reason about near equivalent scenarios based on trends and on companies' experience and their empirical evidences.

Six dimensions of analysis have come up after that, which included the relevance of considering multi-criteria decision-making. They are described below.

3.1.1 Quality of Services (QoS)

QoS is a non-functional dimension defined as the ability to provide different priority to applications, users, data flows, or to guarantee a certain level of general quality. QoS is the largest used technical dimension to select software services and cloud providers, being also considered as a reference for setting up SLA (*Service Level Agreement*). There is a direct relation between SLA fulfillment and provider's reputation [10].

The list of QoS attributes varies a lot in the literature. The *Service Measurement Index* (SMI) initiative has elicited the 33 most relevant QoS attributes for SaaS/services- and cloud-based scenarios [20] (Figure 1), extending a bit and adapting the attributes defined in ISO/IEC 25010.

– Adaptability	– Recoverability	– Security management
– Extensibility	– Stability	– Accessibility
– Flexibility	– Serviceability	– Learnability
– Customizability	– Robustness	– Commonality
– Scalability	– Accuracy	– Multi-tenancy
– Changeability	– Efficiency	– Operability
– Composability	Functionality	– Agility
– Availability	– Response time	– Assurance
– Maintainability	– Suitability	– Performance
– Reliability	– Data integrity	– Security
– Resilience/fault tolerance	– Data privacy	– Usability

Fig. 1. SMI QoS attributes

3.1.2 Reputation

Reputation can be considered as a collective measure of trustworthiness calculated based on the referrals or ratings from members of a community. It can be measured in different ways considering different aspects, usually fed by personal experiences from users [21]. In business, it is basically interpreted as the higher the company's or the service's reputation the higher is its trustworthiness.

3.1.3 Trust

Trust represents the confidence level about an entity, which can be measured as a numerical value and transformed into indicators for decision-making. In business, it is basically interpreted as the trustier a company the less risky it is. Reputation systems are typically used to manage and to establish trust [22].

In the context of networked companies, Msanjilla *et al.* (2007) [23] has created a comprehensive framework to calculate trust by computing five dimensions: financial, organizational, social, technological and behavioral. They also pointed out that trust can be established between the trustor and the trustee, but it can also involve a third part and/or a regulatory institution, as a recognized intermediate entity to certify trust.

3.1.4 Development Quality

This criterion refers to the need of assuring that the potential evocable services are reliable as well as trustworthy and capable to attend the SLA requirements of different software compositions from their many client systems in different tenants [24].

The quality of a software is directly related to the quality of the development process [24]. Sun *et al.* (2012) [25] has highlighted the usual lack of information about the quality of the offered services, creating obstacles for their larger use by customers.

In order to provide higher levels of services quality, SaaS/services providers need to constantly improve their maturity [26]. Organizational maturity is the extent to which it explicitly and consistently deploys processes that are documented, managed, measured, controlled, and continually improved, indicating the organizational path of process improvement [26]. IBM, for example, has developed a services maturity model (called *OSIM* [5]) to assess services and SOA adoption, and to create blueprints to increase business and IT flexibility. Cancian *et al.* (2020) [19] have developed a 5-layer maturity model based on ISO 15504 developed for SaaS providers in collaborative networks.

[5] https://www.ibm.com/developerworks/library/ws-OSIMM/

3.1.5 Governance

In the IT context, governance can be generally defined as the processes to assure the proper use of IT by organizations to help them reaching its objectives[6].
In software services, many works come from the SOA (*Service Oriented Architecture*) area [27]. In essence, the more solid and comprehensive governance processes are implemented in an organization the more trustworthy the general quality of its services tends to be. IBM, for example, defines in its SOA governance model[7] 10 aspects to be governed throughout 26 processes, grouped into 4 categories. Another important work is the *SOA Governance Reference Model (SGRM)*[8], a generic process model used as a baseline to tailor SOA governance models for given organizations.

3.1.6 Collaboration

Collaboration in business can be defined as a working practice by which organizations work on a common purpose to achieve planned benefits [1]. In the context of CN, Correa *et al.* (2014) [17] have devised a model that calculates how risky a given partner is to be added to a given VO based on, among other elements, its successful experience in past VOs as well as when working together with some given partners.

In the context of SOA, this refers to how complex the integration of two heterogeneous services can be (as they should e.g. communicate and exchange documents), or how easier this can be given that some IT providers may have proper APIs to interoperate with each other thanks to existing business partnerships [2].

3.2 Multi-Criteria Methods

Decision-making problems generally imply the selection of the best compromise solution. Besides the values calculated using diverse criteria, the final decision also depends on the decision maker's preferences and underlying policies not always formally calculated or capable to be explicitly expressed as decision criteria [28].

The literature presents several multi-criteria methods, being the *Analytic Hierarchy Process (AHP)* one of the most used ones. AHP allows the combination of quantitative and qualitative data, requiring a judgement about the relative importance (in terms of weights) between one criterion to another [29].

In services discovery, this means calculating a given value via weighting the criteria dimensions to further rank the most suitable services [18].

4 The Proposed Services Dynamic Discovery Model

This section presents the proposed model and multi-criteria algorithm for services dynamic discovery. Figure 2 illustrates the general view of the model.

[6] *COBIT* and *ITIL* are two of very recognized IT governance models for general software and infrastructures.
[7] https://www.ibm.com/developerworks/library/ar-servgov/index.html
[8] https://www.opengroup.org/soa/source-book/gov/p4.htm

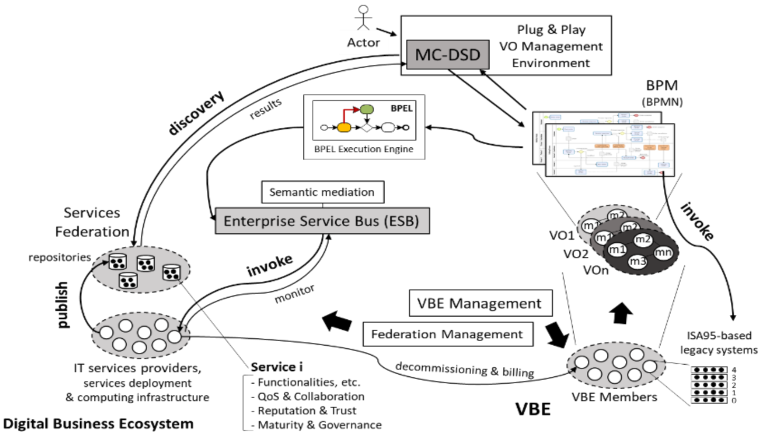

Fig. 2. Basic view of the dynamic discovery model

In general terms, the *Multi Criteria Dynamic Service Discovery* (MC-DSD) acts as a broker, doing a kind of e-procurement, bridging web services sellers and buyers within a digital marketplace. It corresponds to an improvement of a static service discovery module developed in a previous work [30], whose focus was on supporting a seamless ESB-based plug and play integration and semantics interoperation of VOs throughout their life cycle, adopting BP reference models. The whole model also joins the contributions developed for dealing with the VBE/VO management and governance [1,31], federation management [5,32], BPM-SOA integration [2,9], and resilience / fault-tolerance support in the formed services composition when a VO is created [12].

The model works under the *publish-discovery-invoke* SOA reference architecture [10]. In the left side of the figure, the interested software services providers, as members of open and disparate IT ecosystems, prepare their IT assets in way they can be properly published using specific ITs in specific services repositories. Different deployment architectures can be applied for that. A set of metadata has to be added upon each published service, being some data given by the respective provider (e.g. services' functionalities), and some other data maintained and provided by the own Federation (e.g. services' reputation). The federation manages the performance of every provider and single service, updating the respective meta-data accordingly. The federation has a sort of processes used to manage itself and its members [5,32].

The right side shows the VBE and its members (enterprises, industries, etc.), each one having its legacy systems, usually organized according to the ISA-95 layered model. One of the main goals of a VBE is to breed and launch VOs. VOs are mostly created as a response to business demands, and they have a life cycle [1]. The profile of the VO members *m* and the required BP depend on the unique business demands' characteristics. From the dynamic services discovery point of view, the goal is to find the very right services for the diverse BPs. BPM (*Business Process Management*) is a modern approach to help linking business modeling and IT systems [2,9]. *BPMN* has become a standard *de facto* modeling language and different standards for BP models may be adopted, as *RosettaNet* and *EDIFACT. UBL*[9] has been used in this work [2,30].

The MC-DSD allows different actors to tune the supported selection criteria to guide the discovery algorithm.

An internal engine takes every single BPs' activity and triggers a discovery action against the services federation, trying to find out the published services that fit the chosen requirements, bringing their references back to MC-DSD. It, in turn, selects the most suitable ones and instantiates the BP models. A *BPEL* file is generated as a result. Regarding that services are usually very different from each other in terms of implementation technologies, formats and semantics, this file is sent to a message bus, which is an ESB with semantics mediation capabilities that makes the invocation itself to the selected services. Services execution and SLA fulfillment are monitored so that they can be replaced in the case of some fault. All this is very explained in [2,12,30].

As VO members finish the execution of their BPs and hence the usage of the involved services, services are decommissioned, and the respective billing is generated regarding the SaaS mode and signed contract [12].

4.1 Discovery Model Rationale

A set of design principles has been adopted in the devise model:

i. System engineering: the model encompasses different systems from different organizations, each one having their business strategies, IT approaches, life cycles, governance models, and functional and legal responsibilities. This also means that both providers and consumers (i.e. the VO members' systems) can have very different levels of SOA/IT maturity and governance;
ii. Software engineering: providers should consider their services as loosely coupled, interoperable, and non-intrusive entities that, when orchestrated to support the VO operation, will form an integrated business service;
iii. Providers' services represent just one part of the required functionalities of the VOs' BPs. The other part is done by the own VO members' internal systems. VO members can also offer their shareable services via the Federation;
iv. Hybrid integration strategy: *Top-down* - providers can adopt different BP reference models to drive their services' functionalities design and granularity definitions. *Bottom up* - providers can use different implementation tenants and implementation technologies, exposing services via wrappers. A service-bus approach (via an ESB) is used to mediate interoperability. It can be created per VBE, per VO or per Federation, and be deployed in their local servers or in a cloud. Each provider can offer its services under variable business models, versioning and services' lifecycle management, deployment topologies, redundancy policies, and security schemas;
v. Once a service is chosen by MC-DSD, it gets responsible for the entire operation related to the BP's activity it was bound to. The service can do all the needed operations on its own or, complementarily, invoke other providers' services;
vi. The way the VBE and the Federation / Ecosystem handle their members' and services' general performance is up to each one, regarding their bylaws and governance models. For instance, whilst the VBE should define its models (performance, trust, etc.) and manage, supervise, and keep up-to-date the related data upon its members as VOs are executed, the Federation should do the same as for their services and respective providers. Providers should publish their services' metadata and usage costs following the federation's information model. All these data are the ones used by MC-DSD for discovery;
vii. There is a meta (implicit or explicit) governance model, which harmonizes the different working principles of each model's entity to guarantee the whole model operation.

[9] https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=ubl

4.2 The MC-DSD Algorithm

The multi-criteria algorithm considers services' costs and the six dimensions described in section 3: services' QoS and collaboration; providers and services' reputation; providers and services' trustworthiness; services' quality of software and providers' processes maturity; and providers' governance. It is assumed that the classical selection criteria (section 2.1) and the providers' and services' collaboration levels are covered by those dimensions.

The model is generic. Besides the services' functionalities and their interfaces' information, the values and underlying information model each of those dimensions are expressed by the VBE, Federation and providers depend on each model "instantiation" case. Proper IT and tools should be used to model and implement all that[10]. Regarding this proof-of-concept work, it has been instantiated in the implemented prototype (see section 5) as follows, also considering the targeted IT cluster's opinions:

- *QoS*: the 33 aspects proposed in [10,20] were used, being expressed as numbers, considering their intrinsic scale. QoS is the basis for the dynamic SLA establishment [12] and also considers services performance ('collaboration') in past VOs;
- *Trust* and *Reputation*: the 5 dimensions proposed in [23] were used, measuring providers' trustworthiness as well as providers' and services' reputation. They are expressed in a scale varying from 0 (very low) to 10 (very high);
- *Cost*: it directly indicates the maximum acceptable cost to access/use the service, being expressed as a numerical value. It can also indicate a minimum cost, if it was the case;
- *Quality* and *Maturity*: the 5 levels proposed in [19] (and based on CMMI and ISO 15504) were used to express services' quality (in terms of capability) and provider's maturity (collaborative- and SaaS-based processes);
- *Governance*: the 7 levels proposed in [22] were used. In this case, the values are binary, indicating if the company supports it (1) or not (0).
- *Services functionalities*: they are expressed using the UBL BP reference model (as in [2]).

The algorithm is executed in seven steps. It is very similar to the ones applied in several related works, although varying in terms of selection criteria and the ecosystem environment it should work in:

For each VO, for each global BP, for each (sub)BP, for each activity, do:
Specify and configure selection criteria
Search for services in the Federation that fit the selection criteria
Apply a multi-criteria method
Rank the found services and select the best one
Bind the selected service to the respective BP's activity
Repeat until all activities have a bound service

In order to easy the algorithm configuration, user interfaces can hide some details, aggregate dimensions and/or group those values into high-level categories. For instance, instead of indicating numerical values for every single different 33 QoS aspects, (trained) users can just choose for *low, medium, high*, or even indicate that the given aspect is *irrelevant* to be considered in the search. The supporting system(s) should then make a link between such words/specification with the real numbers stored in the related information systems. For example, assuming a scale from 80% to 99.99%

[10] *OASIS*, *Mozilla* and *W3C* are examples of organizations that have been defining standards and providing some open source supporting tools.

of *service availability*, the internal convention can stablish that it is *low* if the value is between 80% and 90%, *medium* if 90.1% and 95%, and *high* if higher than 95%.

The discovery algorithm will then look for services whose availability fits the given range. In the hypothetical governance model, it would be considered that services whose mean *availability* in normal conditions is lower than 80% are not allowed to be published and hence offered by the Federation.

Regarding the large numbers of items to be configured in a search for every single BP's activity of the global VO's BP, companies (each VO member) can have pre-configurations set up as by default (in general, per type of business, etc.) so that the discovery can run completely automatically as VOs get into execution; or users can only modify some items for very particular services' needs. Preliminary rougher searches may be done to better tune the chosen criteria and their parameters.

In the case no services are found out as the result, the search configuration should be redefined (via relaxing some aspects) and executed again. In the very worst case, when no services are found out after many attempts, the missing service(s) should be developed (by some VO member or by some ecosystem's provider), or the VO should find another way (i.e. another BP design) to accomplish the respective business.

Once having at least one service found out for the involved BP's activities, the next main step refers to weighting the degrees of importance between the chosen criteria.

In a large ecosystem composed of potentially hundreds of services providers, it is expected that several services can fit the search criteria for each BP's activity. The problem is then to select which service will be elected as the most suitable one. The essence of multi-criteria methods is helping in this, also based on the premise that it is basically impossible to find a given service that is the best in all the chosen criteria. Therefore, once many services are returned, it is necessary to compare them with each other, which is done via indicating priorities among the chosen selection criteria.

This is done at two levels: intra-criterion and inter-criteria. The first one is basically used for QoS, where the user should specify his preferences. For example, he could indicate that *availability*, *performance*, and *efficiency* should have higher priority than *security* and *assurance*. In the same way, in the inter-criteria level, he could indicate that *provider's governance* should have less priority than *service's costs*. Here again some pre-configurations can be set up as by default.

The AHP multi-criteria method [29] has been used. All those criteria should be compared against to each other to identify which criteria have higher priorities. A matrix is formed with the chosen criteria, and the found services are evaluated based on the defined priority. A subset of the best ranked services is kept. The best one is directly bound the BP's activity, and the remaining ones are kept for future usage.

5 Prototype & Results

Although the resulted outcome and focus of this research is represented by a qualitative model, a prototype has been implemented to generally evaluate its feasibility. This section summarizes that.

As already mentioned, the devised MC-DSD method is an improvement of the service discovery module developed in a previous and complex integration work [30], which only considered some QoS attributes and that were treated as having the same priority in the final services selection. New dimensions of analyses have been added (as described in section 3.1) and a multi-criteria method (section 3.2) has been used to define the different degrees of importance these dimensions have one to another.

The previous prototype [30] were partially reused and adapted to the devised model. Hundreds of services were generated and published adopting the UBL business process reference model as a basis, simulating that all ecosystem's software service providers and VO members would have prepared (tenant wrapping) their services following its specification (in terms of functional requirements, arguments, and data/document types), although with different granularity and implementing technologies. This and other adopted assumptions were explained in section 4.1.

A hypothetical VO was created to produce a given product and the related tasks were distributed among the chosen partners. According to the UBL model, diverse BPs would be involved in this transaction, including the *'Ordering Process'* (one of the 68 UBL's BPs). Due to space restrictions, Figure 3b only shows one part of it (modeled in BPMN), covering the acceptance and rejection of the business request. Many BPs will be normally involved in a VO regarding its full life cycle and the BP's needs of the customer order in place. From the execution point of view, the way the model treats one VO and one UBL BP is exactly the same than as for many VOs and multiple BPs.

VBE's partners information (see section 4.1) have been previously stored in its database (implemented using the *SQL Express* tool). Services repositories were implemented using the *UDDI*, from W3C, using *JUDDI* tool. Services' information is registered in the *tModel* UDDI's structure in *XML*, being partially fed by the respective provider and partially fed by the Federation or VBE. Figure 3a presents an excerpt of the *tModel* of a given service, showing the service's reference (for its future invocation) and the meta data and respective values e.g. for its *reputation* and *trust*.

Fig. 3. (a) Example *tModel* (b) UBL *Ordering Process* BP

For each BP's activity a configuration for services discovery is done (Figure 4a). In this implementation, the user does that via scales instead of via absolute numbers. Before or after that the user sets up the degree of importance among the chosen criteria. The discovery itself is then performed and the AHP method is applied upon them regarding those criteria and indicated weights. A ranking is created once services are retrieved (Figure 4b). The first ranked service is considered as the most suitable one for the *place order* activity and it is automatically bound to it.

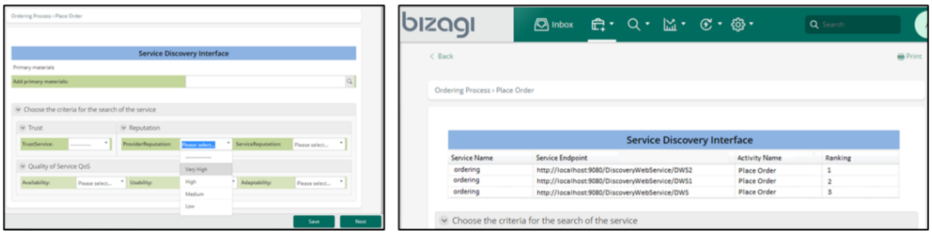

Fig. 4. (a) Discovery configuration (b) Services ranking

6 Final Considerations

This paper has presented results of an ongoing research that aimed at devising a more comprehensive set of selection criteria for dynamic (web) services discovery that better fits a scenario of open ecosystems of software providers, including collaborative and virtual organizations.

The model is flexible in a way that more criteria can be added as new scenarios arise up. It was built as a proof-of-concept considering a local cluster of IT-based SMEs. The multi-criteria approach allows companies tuning the service discovery algorithm according to their current business needs, governance principles, and preferences.

A system prototype was implemented using open source tools and IT standards, in a controlled environment. It can be said that the federation concept and related digital platform represent an approach towards Collaboration 4.0.

According to the IT cluster, and despite the several adopted assumptions, results showed that the model is promising and seems to be comprehensive enough for dealing with evolving open ecosystems of SaaS-based providers. Foreseen obstacles identified by the cluster for the model adoption include: it can take a long time given the many issues to handle; demands some trust building; and it requires investments on governance, IT and human resources. It also calls for a new mindset and much higher complexity on how building and integrating disparate computing systems.

Next main steps of this research include: the consideration of the IT cluster's services to test the discovery model in a near-real business environment; and to define better strategies to decrease the searching algorithm's computing complexity.

Acknowledgements. This work has been partially supported by CAPES - The Brazilian Agency for Higher Education, project PrInt CAPES-UFSC "Automation 4.0".

References

1. Camarinha-Matos, L. M., Fornasiero, R., Ferrada, F.: Collaborative Networks - A Pillar of Digital Transformation. *Applied Sciences*, 9(24) (2019).
2. Bezerra, R. O., Rabelo, R. J., Cancian, M. H.: Enhancing Network Collaboration in SOA Services Composition via Standard Business Processes Catalogues. Proceedings 18th IFIP Working Conference on Virtual Enterprises, pp. 421-431. Springer (2017).
3. Romero, D., Vernadat, F.: Enterprise information systems state of the art: past, present and future trends. *Computer in Industry*. N 79, pp. 3-13 (2016).

4. Rabelo, R. J., Gusmeroli, S.: The ECOLEAD collaborative business infrastructure for networked organizations. In *Methods and Tools for Collaborative Networked Organizations*, pp. 451-462. Springer (2008).
5. Cancian, M. H.; Rabelo, R. J.; Wangenheim, C. G. V.: Collaborative business processes for enhancing partnerships among software services providers. *Enterprise Information Systems*, v. 9, pp. 1-26 (2015).
6. Chou, T.: *The End of Software: Finding Security, Flexibility, and Profit in the on Demand Future*, Sams Publishing (2004).
7. Gansky, L: *The mesh – why the future of business is sharing*. Penguin Books (2010).
8. Sitek, P., Gusmeroli, S., Conte, M., Jansson, K., Karvonen, I.: *The COIN Book - Enterprise Collaboration and Interoperability*, COIN IP EU Project (2010).
9. Souza, A. P., Rabelo, R. J.: A Dynamic Services Discovery Model for better leveraging BPM and SOA integration, *Int. J. Information Systems in the Service Sector*, 7(1), p. 1-21 (2015).
10. Khanjani, A., Nurhayati, W., Rahman, W.: Saas Quality of Service Attributes. *Journal of Applied Sciences*, 14(24), pp. 3613-3619 (2014).
11. Ardagna, D., Baresi, L., Comai, S., Comuzzi, M.: A Service-Based Framework for Flexible Business Processes. In: *Software*, IEEE vol. 28(2) pp. 61-67 (2011).
12. Bezerra, R., Rabelo, R. J., Cancian, M. H.: Supporting SOA Resilience in Virtual Enterprises. Proc. 19th IFIP Working Conference on Virtual Enterprises. Springer, pp. 111-123 (2018).
13. Moghaddam, M.: Simultaneous service selection for multiple composite service requests: A combinatorial auction approach. *Decision Support Systems*, 81-94 (2019).
14. Obidallah, W. J., Raahemi, B.: A survey on web service discovery approaches. Proc. 2nd International Conference on Internet of Things, Data and Cloud Computing, pp. 1-8 (2017).
15. Camarinha-Matos, L.M., Afsarmanesh, H., Koelmel, B.: Collaborative Networks in Support of Service-Enhanced Product. In: Adaptation and Value Creating Collaborative Networks. PRO-VE 2011. IFIP Advances in Information and Communication Technology, vol 362. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-642-23330-2_11 (2011).
16. Osorio, L., Afsarmanesh, H., Camarinha-Matos. L.: A Service Integration Platform for Collaborative Networks**,** *Studies in Informatics and Control*, 20 (1), pp. 19-30 (2011).
17. Correa, O. A., Rabelo, R. J., Vieira, R. G., Fiorese, A.: A Risk Analysis Method to Support Virtual Organization Partners' Selection. In: Proceedings 15th IFIP Working Conference on Virtual Enterprises, Springer, pp. 597-609 (2014).
18. Bandara, K. Y., Wang, M.: An extended ontology-based context model and manipulation calculus for dynamic Web service processes, *S.O.C. and Applications*, pp. 87-106 (2015).
19. Cancian, M. H., Hauck, J. R.: Towards a capability and maturity model for Collaborative Software-as-a-Service. *Innovations in Systems and Software Engineering*, 1, pp. 1-25 (2020).
20. Ferretti, S., Ghini, V., Panzieri, F., Pellegrini, M.: QoS-aware clouds. Proceedings IEEE 3rd International Conference on Cloud Computing, pp. 321-328 (2010).
21. Resnick, P., Kuwabara, K., Zeckhauser, R., Friedman, E.: Reputation Systems. *ACM Communication*, 43, pp. 45-48 (2000).
22. Sabater, J., Sierra, C.: Review on computational trust and reputation models. *Artificial Intelligence Review*, 24, pp. 33-60 (2005).
23. Msanjila, S. S., Afsarmanesh, H.: Modeling Trust Relationships in Collaborative Networked Organizations. *International Journal of Technology Transfer and Commercialization*, Inderscience, 6(1), pp. 40-55 (2007).
24. Beecham, S., Hall, T., Britton, C.: Using an expert panel to validate requirements process improvement model. *Journal of System Software*, 76(3), pp. 251-275. (2005).
25. Sun, L.; Singh, J.; Hussain, O. K.: Service level agreement (SLA) assurance for cloud services: a survey from a transactional risk perspective. Proceedings 10th ACM International Conference on Advances in Mobile Computing Multimedia, pp. 263-266 (2012).

26. Knoke, B.: Innovation Capability Maturity within Collaborations. Proceedings 25th International Conference on Advanced Information Systems Engineering, pp.7-11 (2013).
27. Hojaji, F., Shirazi, M. A.: A Comprehensive SOA Governance Framework based on COBIT. Proceedings 6th World Congress on Services, pp. 407-414 (2010).
28. Daghouri, A., Mansouri, K.: Multi Criteria Decision Making methods for Information System Selection - A Comparative Study. Proceedings International Conference on Electronics, Control, Optimization and Computer Science, pp. 1-5 (2018).
29. Saaty, R. W.: The analytic hierarchy process - what it is and how it is used. *Mathematical Modelling*, 9(3-5). Springer, pp. 161-176 (1987).
30. Mendez, J. D., Rabelo, R. J., Cancian, M. H.: A Plug and Play Integration Model for Virtual Enterprises. Proc. 19th IFIP Working Conf. on Virtual Enterprises. pp. 312-324 (2018).
31. Costa, S. N., Rabelo, R. J., Romero, D.: A Governance Reference Model for Virtual Enterprises. In: Proc. 15th IFIP Working Conference on Virtual Enterprises, pp. 60-70 (2014).
32. Cancian, M. H., Teixeira, C. P., Rabelo, R. J.: Supporting Business Processes for Collaborative Alliances of Software Services Providers. Proceedings 16th IFIP Working Conference on Virtual Enterprises. Springer, pp. 467-478 (2015).