A Survey of Fuzzy Service Matching Approaches in the Context of On-The-Fly Computing

Marie C. Platenius[*] , Markus von Detten, Steffen Becker, Wilhelm Schäfer
Software Engineering Group
Heinz Nixdorf Institute
University of Paderborn, Germany
m.platenius@upb.de

Gregor Engels
Database and Information Systems
Department of Computer Science
University of Paderborn, Germany
engels@upb.de

ABSTRACT

In the last decades, development turned from monolithic software products towards more flexible software components that can be provided on world-wide markets in form of services. Customers request such services or compositions of several services. However, in many cases, discovering the best services to address a given request is a tough challenge and requires expressive, gradual matching results, considering different aspects of a service description, e.g., inputs/ouputs, protocols, or quality properties. Furthermore, in situations in which no service exactly satisfies the request, approximate matching which can deal with a certain amount of fuzziness becomes necessary. There is a wealth of service matching approaches, but it is not clear whether there is a comprehensive, fuzzy matching approach which addresses all these challenges. Although there are a few service matching surveys, none of them is able to answer this question. In this paper, we perform a systematic literature survey of 35 (out of 504) service matching approaches which consider fuzzy matching. Based on this survey, we propose a classification, discuss how different matching approaches can be combined into a comprehensive matching method, and identify future research challenges.

Categories and Subject Descriptors

H.3.5 [**Information Storage and Retrieval**]: On-line Information Services

Keywords

Service Matching, Fuzzy Matching, Gradual Matching Results, On-The-Fly-Computing, Survey

[*]This work was supported by the German Research Foundation (DFG) within the Collaborative Research Center "On-The-Fly Computing" (CRC 901).

1. INTRODUCTION

The increasing complexity of software systems is a known problem in software engineering [40]. Therefore, in the last decades, development turned from monolithic software products towards more flexible, component-based and service-oriented solutions. For example, in paradigms like Service-Oriented Computing or On-The-Fly (OTF) Computing [54], software components are readily deployed for use by third parties and can be provided in form of services on world-wide markets (Software-as-a-Service) and customers request these services or service compositions. In OTF Computing these markets are assumed to be large, dynamic and heterogeneous, serving many different domains.

However, assuming that the variety of provided services that are candidates to fulfill a requirement, it is not only necessary to match services, but also to find the most appropriate one. This can be accomplished by ranking the service candidates regarding their suitability [51]. Such a ranking requires matching approaches to determine how much a service matches the request instead of just returning "does match" or "does not match". Thus, matching approaches have to be capable of *fuzzy matching*, i.e., determining approximate matches which can slightly deviate from an exact match, and to quantify this deviation. Such deviations can have different reasons, e.g., imprecise descriptions of the request or the service.

Nowadays, there is a wealth of different service matching approaches which offer a large bandwidth of matching functionality. Especially the extent to which fuzzy matching is supported varies. However, due to the huge number of matching approaches, it is unclear whether an adequate fuzzy matching approach, that satisfies certain requirements, exist. In our paper, we are especially interested in requirements derived from OTF Computing.

The few existing surveys in the area of service matching are not sufficient to answer this question because they are limited to certain types of matching approaches (e.g., semantic web service matching) and do not cover the OTF Computing requirements. Especially, they do not explicitly address fuzziness.

In this paper, we survey available service matching approaches with respect to the requirements of OTF Computing. Therefore, we perform a systematic literature review of 35 (out of 504) service matching approaches that consider – at least to a certain extent – fuzzy matching. According to the challenges mentioned above, the literature review is driven by three main research questions:

Q1 To which extent is fuzzy matching considered in current service matching approaches?

Q2 Are there service matching approaches that, in addition to fuzzy matching, satisfy all requirements of On-The-Fly Computing?

Q3 If the answer to Q2 is "no": what is missing in order to meet these requirements? Which challenges should be addressed in future research?

Based on this review, we propose a classification and discuss existing matching approaches regarding these questions.

In the next section, we introduce our definition of service matching and describe the requirements OTF Computing has for service matching approaches. Section 3 illustrates the procedure we used to conduct the survey. The classification of the selected approaches is presented in Section 4 and discussed in Section 5. Section 6 deals with related work. The paper is concluded in Section 7.

2. MATCHING REQUIREMENTS

In this paper, we define service matching as follows: Service matching compares the specification of a service request to the specification of a provided service in order to determine their interoperability. This is done by checking different *matching conditions* based on different kinds of available information, which we call *aspects*. For example, for the aspect "inputs/outputs", one condition for each input and one condition for each output could be checked. Service requests can either stem from service discovery (i.e., the user requests a specific service) or from service composition (i.e., the service is requested by a human or composition algorithm in order to be incorporated into a composite service). However in both use cases, service discovery and service composition, the requests are similar and thus, they can be reduced to the same matching problem.

According to the OTF Computing vision described above, service matching approaches in the context of On-The-Fly Computing underly several requirements:

R0: Automation OTF Computing tries to minimize human interaction in service discovery and composition. This implies that service matching approaches are required to be as automated as possible.

R1: Structured Specifications In order to automatically compose two services, some essential information about the structure and semantics of their interfaces has to be taken into account. Structured specifications explicitly allow for accessing different aspects, like input and output types in a machine-readable form. Therefore, a matching approach has to be able to match structured specifications. In contrast, matching approaches working with unstructured specifications, e.g., plain text, simple lists of keywords, or logical formulas without further classification, are not useful in OTF Computing.

R2: Comprehensiveness The matching of structural aspects, e.g., datatypes of inputs and outputs, is not sufficient to achieve reliable matching results. The behavior of a service also has to be taken into account. Thus, many matching approaches consider aspects like ontologies, pre- and postconditions, and protocols, describing different details of a service. Furthermore, quality of service (QoS) becomes more and more relevant and needs to be matched as well. Therefore, in OTF Computing, we aim at using a matching approach that matches comprehensive service specifications describing as many aspects as possible.

R3: Efficiency Efficiency is an important issue in matching, especially in large-scale environments [1]. Since OTF Computing deals with world-wide, dynamic markets, efficiency is an important requirement for matching approaches to be applied in OTF Computing.

R4: Fuzzy Matching In order to select the best service from a set of potentially matching services, matches need to be ranked, i.e., gradual matching results have to be determined. This requires a matching approach to not only determine exact matches, but also situations, in which services only match to a certain extent, i.e., fuzzy matching is required. In particular in OTF Computing, where the dynamic market situation leads to an uncertain environment, fuzzy matching is essential.

There are some interdependencies between these requirements. While R1 is essential for the fulfillment of R2, R2 and R3 form a trade-off: On the one hand, a reliable matching of comprehensive service specifications with a large amount of different information may impede the decidability and thereby the effiency and automation. On the other hand, the more efficient a matching aims to be, the less information of the services can be considered. Furthermore, the more fine-grained matching degrees are to be determined (R4), the more information (R2) and time (R3) is needed. Trade-offs like these complicate the fulfillment of all requirements by one single matching approach.

3. SURVEY PROCEDURE

Driven by the research questions introduced in Section 1, we chose the following procedure.

We executed our survey according to Kitchenham's well-established guidelines for systematic literature reviews in software engineering [29], aiming to achieve objective, unbiased, and reproducable results. The literature search was conducted between June 2012 and January 2013 and we used Google Scholar, ACM, IEEE Xplore, and Science Direct as search engines.

We performed a keyword-based search in combination with "snowballing" (i.e., following relevant outgoing references of the discovered publications). The relevance of a publication was estimated using different well-defined criteria in different steps of the following process:

1. **Initial set of service matching publications:** First, we selected publications based on the occurrence of a combination of defined terms in their title. The term combination is composed from a set of substantives denoting entities that can be matched (e.g., "Service", "Component", "Interface", ...) and a set of terms implying some kind of matching activity (e.g., "Matching", "Discovery", "Interoperability", "Comparison", ...).

A publication is declared as relevant, if at least one of the terms from both sets appears in the title (e.g., "Service" + "Matching", "Component" + "Matching", or "Component" + "Interoperability"). Using this strategy, we selected 504 publications. Due to space restrictions in this paper, the complete list of selected publications, as well as the complete lists of used terms are provided on our website [45].

2. **Refined set of fuzzy matching publications:** In the next step, we excluded a large part of the initial set of publications based on the following criteria: a) approaches based on unstructured service specifications (R1), b) manual matching approaches (R3), c) approaches that do not support fuzzy matching (R4), and d) approaches that are not related to service matching according to our definition, i.e., false positives with regard to the term-based search used in Step 1 that we manually detected with our own expertise. This analysis has been performed based on the abstracts and author-defined keywords first. After that, we filtered out the remaining publications based on the same criteria but considering the full-text of the papers. Furthermore, publications describing the same approach were grouped in this step. Using this strategy, this resulted in 74 matching approaches which are highlighted with colored rows in the list on our website [45].

To handle this still large set of approaches with regard to space restriction in this paper, we added two further selection criteria:

- Assuming that the considered approaches should be mature and provide an evaluation, we only selected conference papers, journal papers, or book chapters. Thereby, we tried to only select high quality papers and to exclude papers describing approaches in the very early stages of research, e.g., workshop papers. We performed this step manually and thereby excluded 9 publications.
- To get an overview of the current state of the art, in this paper, we decided to focus on the most recent approaches. Thus, we only selected approaches presented in publications since 2008 and thereby excluded 30 publications.

These decisions lead to a final number of 35 approaches presented in the classification in this paper.

Note that our search strategy obviously may lead to the unwarrented exclusion of relevant papers (false negatives). Especially by excluding publications from before 2008, we may lose some interesting research in the area of software component matching, e.g., the approach presented by Zaremski and Wing [59] which supports "relaxed matching". However, we expect valuable research results coming from earlier publications to be also reflected in the more recent approaches. We further discuss the conclusion validity in Section 7.

4. CLASSIFICATION

We classified the selected fuzzy matching approaches according to different aspects of the services' specifications that are matched: Ontologies, Input/Outputs, Pre-/Post-conditions, Protocols, and QoS, as depicted in Table 1. This structure reflects our requirement R2 and is based on related classifications [5, 18, 31]. Ontology matching means that ontologies are leveraged for matching. There are different possibilities to use ontologies, but typically they represent domain knowledge which can be reused during matching one of the other aspects, e.g., inputs/outputs. Input/Output matching refers to matching the data types the service expects as an input and delivers as an output. Pre- and postcondition matching takes into account further assertions about a service's functionality. Protocol matching checks the required and provided sequences of ingoing and outgoing messages, i.e., service invocations. Unlike the other aspects, QoS matching takes into account non-functional properties of the service, e.g., response time or availability.

With respect to these aspects, we evaluated the approaches using three grades: "+" (matching considers this aspect), "o" (matching partly considers this aspect), "-" (matching does not consider this aspect).

The comparison of the different approaches is complicated because different authors use different terminologies according to the underlying specification languages (shown in the last column of Table 1). In order to allow for a comparison of the approaches, we adjusted the relevant terms. For example, we interpreted "Postconditions", "Effects", and "Results" as synonyms and only use the term "Postconditions" from now on.

Some approaches are independent of a concrete specification language and did not mention any specification language in their publications. Thus, there are empty fields in the last column.

In most cases, the matched aspects could be clearly distinguished between not supported (-) and supported (+). There is only one exception: While many approaches use ontologies to leverage domain knowledge, the approach presented by Liu et al. [37] is not explicitly based on an ontology. Instead, they use a web search engine for the same tasks.

5. DISCUSSION

In this section, we discuss our findings in more detail. First, we will discuss the results that evolve from the aspect-based classification. After that, we will analyse how fuzziness is considered. At the end of this section, we summarize the main issues with respect to the research questions and derive future research challenges in service matching.

5.1 Discussion of aspects

As Table 1 illustrates, our selection contains no matching approach that takes all five aspects into account, which answers Q2 because OTF Computing requires a comprehensive matching approach that considers all of these aspects. However, most of the selected fuzzy matching approaches go beyond simple input/output matching and combine it with a matching of one of the other aspects. Protocol matching is rather uncommon in the selected set of approaches. Furthermore, some approaches match QoS properties but for the most part, these approaches only focus on QoS and do not or only occasionally consider functional aspects like inputs and outputs.

Nevertheless, already before selecting approaches for further analysis, the enormous set of existing matching approaches is notable. There are a lot of similar approaches solving the same or comparable problems. Many of these approaches overlap in their contributions. Therefore, it seems

Table 1: Aspect-based Classification

	Matched Aspects (-/o/+)					
	Ontologies	**Inputs/ Outputs**	**Pre/Post-conditions**	**Protocols**	**QoS**	**Specification Languages**
Bacciu et al. [2]	-	-	-	-	+	Dino
Bener et al. [8]	+	+	+	-	-	OWL-S
Bellur & Vadodaria [6]	+	+	+	-	-	OWL-S + expr. lang.
Bianchini et al. [10]	+	+	-	-	-	
Brogi et al. [11]	+	+	-	+	-	OWL-S
Cuzzocrea & Fisichella [15]	-	-	-	+	-	OWL-S processes
Corrales et al. [14, 23, 24]	-	+	-	+	-	BPEL, WSCL
Chabeb et al. [13]	+	+	+	-	-	WSDL + YASA
D'Mello et al. [16]	-	-	-	-	+	
Fiadeiro et al. [21]	+	+	-	+	+	SRML
Fenza et al. [20]	+	+	+	-	-	OWL-S
Giunchiglia et al. [22]	+	+	+	-	-	WSDL
HongKang et al. [26]	+	+	-	-	+	
Hao et al. [25]	-	+	-	-	-	WSDL
Jeong et al. [27]	-	+	-	-	-	WSDL
Ke & Huang [28]	+	+	-	-	-	OWL-S
Klein et al. [30]	-	-	-	-	+	
Klusch & Kapahnke [32, 33]	+	+	+	-	-	
Liang & Lam [35]	+	+	-	-	-	
Lin et al. [36]	+	-	-	-	+	
Liu et al. [3, 4, 38, 39]	+	+	+	-	+	WSDL + OWL-S/ WSMO
Liu et al. [37]	o	+	-	-	-	WSDL
D'Mello et al. [17]	+	+	+	-	+	OWL-S
Naeem et al. [41]	-	-	+	-	-	Visual Contracts
Palmonari et al. [43]	+	-	-	-	+	
Peng et al. [44]	+	-	-	-	+	OWL-S
Plebani & Pernici [46]	+	+	-	-	-	(SA)WSDL
Skoutas et al. [49, 50, 51]	-	+	+	-	-	
Spanoudakis & Zisman [52]	-	+	-	+	+	WSDL + WSCL/BPEL
Tran et al. [53]	+	+	-	-	-	SAWSDL
Wan [55]	-	-	-	-	+	
You et al. [56]	+	+	+	-	-	
Yun [57]	+	+	-	+	-	
Zamanifar et al. [58]	+	+	-	-	-	WSDL
Zhang et al. [60]	-	+	-	-	+	WSDL

sensible to put more focus on research about how to combine different approaches and how to aggregate their results in order to achieve a more comprehensive approache without "reinventing the wheel".

Some approaches already go into this direction and aim at extending existing approaches. For example, Bacciu et al. argue that their approach can be adapted to provide fuzzy-valued extensions of non-fuzzy QoS matching methods [2]. Bellur and Vadodaria present a pre- and postconditions matching which is explicitly created to augment existing approaches that only cover matching of inputs and outputs [6]. Furthermore, in the last years, some approaches for "self adaptive" service matching appeared. These approaches use learning techniques in order to discover the most successful way to aggregate the results of several matching algorithms, e.g., [33]. However, they highly depend on an adequate training set given in advance which is not realistic in paradigms that expect very dynamic markets, like OTF Computing. Furthermore, these approaches are limited to adapting the aggregation of results within one aspect and not considering different aspects matched by completely different conditions.

Some approaches consider weights for the different matching results in their aggregation formulas, e.g., [33, 38]. When combining different matching approaches, such a weighting technique could be useful. Weights could either be provided by the requester or by the service specification, or they could be calculated using learning techniques.

However, the aggregation of different results always leads to some loss of information which could be valuable in further processing. For example, some approaches do "composition-oriented matching": if no exact match is available, service compositions are searched or created, e.g., [11] or [41]. If the matching procedure is part of a complex composition process, it would not only be helpful to know some kind of matching degree (e.g., 80%), but also, to know, which conditions exactly caused such a low final score (e.g., a certain output is missing or one of the required QoS values is not met). Brogi et al. [11] solve this issue in their approach by suggesting additional inputs that are missing to get a full match. Another common use case supported by such results is the generation of adapters [12]. Furthermore, such expressive results are not only interesting for further automated processing, but also for a human requester reconsidering his or her requirements and possibilities.

Gradual results and their causes and consequences are further discussed in the next subsection.

5.2 Discussion of fuzziness

While analyzing the selected matching approaches, we paid special attention to how fuzziness occurs in matching (Q1) and how the approaches deal with it, especially in the

Table 2: Fuzziness-based Classification

		Sources of Fuzziness (+/-)			
	Matching Result Degrees	**Incomplete Knowledge**	**Variational Scope**	**Heuristics & Simplif.**	**Calculation Concepts**
Bacciu et al. [2]	continuous	+ (prov.)	+ (req.)	-	FL
Bener et al. [8]	continuous	-	+ (req.)	-	SR, BPGM
Bellur & Vadodaria [6]	combination of 3 subsumption degrees	-	+ (req.)	+	SMT + SR + BPGM
Bianchini et al. [10]	continuous	-	+ (req.)	-	SR
Brogi et al. [11]	- (add. inputs)	-	+ (req.)	-	
Cuzzocrea & Fisichella [15]	continuous	-	+ (req.)	+	GM
Corrales et al. [14, 23, 24]	continuous	-	+ (req.)	+	GM
Chabeb et al. [13]	continuous	-	+ (req.)	-	SR
D'Mello et al. [16]	continuous	-	+ (req.)	-	
Fiadeiro et al. [21]	continuous	-	+ (req.)	-	
Fenza et al. [20]	?	-	+ (req.)	+	FL
Giunchiglia et al. [22]	continuous	-	+ (req.)	+	
HongKang et al. [26]	continuous	-	+ (req./prov.)	-	FL
Hao et al. [25]	continuous	-	+ (req.)	-	
Jeong et al. [27]	continuous	-	+ (req)	-	
Ke & Huang [28]	continuous	-	+ (req.)	-	
Klein et al. [30]	continuous	-	+ (req.)	-	LP
Klusch & Kapahnke [32, 33]	continuous	-	-	+	SR
Liang & Lam [35]	continuous	-	-	+	
Lin et al. [36]	continuous	-	+ (req./prov.)	+	FL
Liu et al. [3, 4, 38, 39]	continuous	-	+ (req.)	-	SR, FL
Liu et al. [37]	continuous	-	+ (req.)	-	BPGM
D'Mello et al. [17]	continuous	-	+ (req.)	-	SR
Naeem et al. [41]	- (partial match)	+ (req.)	+ (req.)	-	GM
Palmonari et al. [43]	continuous	+(req./prov.)	+ (req.)	-	
Peng et al. [44]	continuous	-	+ (req.)	-	FL
Plebani & Pernici [46]	continuous	-	+ (req.)	-	LP + BPGM
Skoutas et al. [49, 50, 51]	- (ranked list)	-	+ (req.)	-	SR
Spanoudakis & Zisman [52]	continuous	-	+ (req.)	+	GM
Tran et al. [53]	4 cont. degrees + match level	-	+ (req.)	-	
Wan [55]	continuous	+ (prov.)	+ (req.)	-	FL
You et al. [56]	continuous	-	+ (req.)	+	
Yun [57]	continuous	-	+ (req.)	-	
Zamanifar et al. [58]	subsumption degrees	-	+ (req.)	+	BPGM + SR
Zhang et al. [60]	- (ranked list)	-	+ (req.)	-	

matching results. Table 2 illustrates the comparison in a second, fuzziness-based, classification scheme.

Matching result degrees.

The first column of Table 2 deals with the result degrees returned by the matching approaches. These degrees denote the final degrees, i.e., the degrees returned by the matching algorithm after termination. Most approaches support continuous values as final matching degrees. These results most often are constituted by aggregating preliminary results for different matching conditions. The approach by Fenza et al. [20] delivers different degrees for the different matching conditions, but it is not documented how the final result is determined. Some approaches do not return values but other kinds of output, e.g., ranked lists [49, 60]. Zhang et al. [60] explicitly discuss that and state that rank positions are better suited than absolute values because (in their approach) the absolute values indicate different features of the services and include different units and range.

In many approaches based on ontologies, gradual results are determined by semantic distances between the concepts the input and output data types refer to. Semantic distances are most often determined based on "kind of" relations in the ontology, i.e., subsumption reasoning. Before aggregating, matching conditions based on subsumption reasoning either result in the common subsumption degrees (exact, plug-in, subsume, fail) (e.g., in [8, 13, 17, 58]) or determine a more expressive score that is able to distinguish between different degrees of subsumption denoting distance of different lengths (e.g., in [38, 50]). Aggregation of such degrees or scores is performed with different strategies, e.g., selecting the minimum value or calculating the arithmetic mean.

The only approach that clearly distinguishes between different groups of matching degrees in the matching result is presented by Tran et al. [53]. Their matching approach returns a match result and a match level. The match result consists of four degrees: Service input fulfilment, request input redundance, request output fulfilment and service output redundance. The match level can be either "precise", "over", "partial", or "mismatch", depending on the match degrees. Using such a structured matching result, they aim at distinguishing between a set of services resulting in the same match level and thereby facilitating the following service selection.

In the following discussion, we will take up the characteristics of matching results with respect to different sources that constitute these results.

Sources of fuzziness.

An interesting finding is that in the different approaches, matching is confronted with different kinds of fuzziness originating in different sources. In particular, we identified three different types of fuzziness sources in service matching (second, third, and fourth column in Table 2), explained in the following:

Incomplete Knowledge: A basic requirement for a matching approach in the semantic web is the ability to deal with incomplete knowledge [34]. Some matching approaches consider incomplete knowledge in terms of expecting incomplete service specifications provided by the service provider. In addition, a requester could also have incomplete knowledge about the requirements he or she wants to specify.

Variational Scope: In situations in which there are few or no perfect matches, the customer may tolerate slight variations from the request. In some cases, a customer's request even may explicitly specify variation possibilities ("I want either this or that") or vague, imprecisely defined terms ("The service should be fast"). In addition, requests could intentionally omit some information ("I do not care how expensive the service is"). Furthermore, a service could allow to be used differently, e.g., the signature or the protocol could contain optional parts that are not essential for the service's main functionality but that simplify or improve its actions.

Heuristics and Simplifications: Fuzziness could also result from the underlying algorithms within a matching algorithm itself. Decisions could be made based on heuristics or on simplifications necessary to keep the approach decidable and efficient, e.g., if the expressiveness of the underlying language hinders easy comparison according to the trade-off described in Section 2. The difference between this type of fuzziness and the variational scope is that in the latter case, fuzziness is created intentionally, assuming that the requester tolerates variations. In contrast, heuristics and simplifications in the underlying algorithm induce fuzziness not explicitly having a customer's request in mind.

Accordingly, fuzziness can be induced by different participants of the scenario: Incomplete knowledge and variational scope come either from the provider or the requester side, and heuristics and simplifications are caused within the matching algorithm itself.

Incomplete knowledge on the provider side can have two different reasons: either the service provider does not (precisely) know some properties of the provided services, e.g., availability [2], or the service provider does not want to expose certain information, as in [55]. Naeem et al. [41] assume incomplete knowledge on the requester side about the data and resources that may be required to satisfy the requester's needs. This problem occurs because the underlying specification language aims at representing very detailed, fine-grained specifications of a service's functionality. From this we derive that, the more fine-grained the service specifications are required to be, the more knowledge is needed from both provider and requester and the more relevant becomes the problem of incomplete knowledge.

However, most approaches do not explicitly consider the case of incomplete knowledge in their matching algorithms. Nevertheless, returning gradual results might implicitely already support this case to some extent because if some information is missing in a service description, this service would not be refused. Instead it would only receive a lower degree. Nevertheless, the requester could not distinguish between the two different reasons of low matching degrees: a) degrees caused in some failed matching conditions because of not matching service properties and b) degrees caused in some failed matching conditions because the required information for a successful match was not available.

However, most often, fuzziness is induced by assuming the requester tolerates variational scope. The reason might be that the requester's requirements are most often imprecise or imprecisely defined. Accordingly, the matching result typically shows how much a service satisfies the request.

However, there are also some exceptions. For example in [25], Hao et al. define a notion of "service importance" stating how often this service is used by other services in service compositions. For the final ranking, this service importance value is aggregated with a value that denotes how much the service fits the request.

Variational scope is - especially for QoS properties - often given by the presence of imprecise terms in either service specifications or requests, e.g., [2, 26, 36, 44, 55]. Lin et al. [36] explicitly discuss the challenge of allowing such imprecise terms in requests and service specifications. Service providers and service requesters might have different opinions on certain fuzzy terms. For example, a service might by "very fast" in the view of a service provider, but a service requester could call the same service "a bit slow". Also different requesters may have different expectations of different QoS levels and so do different providers. The solution presented by Lin et al. is a "Fuzzy Moderator" that coordinates the expectations from both sides by resolving conflicts using a similarity aggregation measure.

In general, variational scope most often appears in QoS or ontological matching (semantic distances, as explained above). However, there are also examples of such fuzziness in matching approaches that match other aspects. For example, Yun [57] determines "weak equivalence" in process descriptions and thereby calculates matching degrees.

Although so many approaches are designed to allow variational scope by the requester, in most cases this requester cannot explicitly specify this scope, i.e., he or she cannot specify which parts of the request are mandatory and which are preferred but optional. In our selection, only two approaches [43, 52] allow this by distinguishing between "hard constraints" and "soft constraints".

Furthermore, there are some examples in which fuzziness is not induced by the requester or provider, but by heuristics. Such techniques are most often used in order to simplify the matching, e.g., as presented in [6], because of the underlying satisfiability problem (SAT) which is NP-Complete. Similarly, in the approach presented by Cuzzocrea and Fisichella [15], as well as in the approach by Corrales et al. [14, 23, 24], graph matching is simplified in order to reduce the computational complexity of the maximum common subgraph isomorphism problem which is NP-hard. In both cases, these simplifications are necessary in order to keep the matching efficient (R3). However, the resulting matching degrees do not reflect the extent of fuzziness which occurs due to this simplifications, but only the resulting similarity of the matched services (the second source of fuzziness, allowing variational scope), concealing the uncertainty of these results caused by the simplifications.

Relating the aspect-based classification to the fuzziness sources, we noted that fuzziness most explicitly occurred in QoS matching. For example, HongKang et al. [26] use fuzziness only for QoS matching, not for the other aspects. The reason could be that QoS properties are by nature "increasing" or "decreasing", as noted by D'Mello et al. [16].

Underlying calculation concepts.

An interesting finding is also the variety of underlying calculation concepts as collected in the last column of Table 2. The abbreviations represent the different calculation concepts: SMT = Satisfiability Modulo Theory; GM = Graph Matching; BPGM = Bipartite Graph Matching; FL = Fuzzy Logic / Fuzzy Set Theory; SR = Subsumption Reasoning; LP = Linear Programming.

From our analysis, we noted some common traits: Many fuzzy matching approaches are based on fuzzy logic, or fuzzy set theory respectively (e.g., [2, 3, 4, 36, 39, 55]). Bacciu et al. [2] argue that fuzzy sets are suited for the specification of requests (if the service requester specifies preferences, e.g., for QoS), as well as for service specifications provided by the service provider (if the provider cannot give exact numbers, e.g., for the QoS properties like availability). These fuzzy logic computations have the advantage that they are not very expensive in terms of computation complexity (R3). In contrast, more complex calculations like graph matching, most often require some simplifications in order to keep the matching approach efficient, as explained above.

5.3 Identified issues and research challenges

In this section, based on the systematic literature survey, we answer the research questions introduced in Section 1 and present future research challenges.

In our discussion, we derived three sources of fuzziness: Incomplete knowledge, variational scope, and heuristics and simplifications. This answers our research question Q1. Our aspect-based classification showed that in our selection, there is no comprehensive approach that supports matching of all five relevant aspects (R1). Since our selection has been constituted with respect to the other requirements from On-The-Fly Computing (R1, R3, and R4), we conclude that there are no service matching approaches that satisfy all requirements and thereby answer Q2 with "no". Referring to the aspect-based classification, especially pre- and postconditions, protocols, and QoS properties are rarely matched in combination with each other, which answers Q3.

In the following, we use the results from our discussion to derive future research challenges (Q3).

Combination of existing approaches: Since we discovered many service matching approaches for the same problems but with different advantages, a promising way to more advanced service matching seems to be the work on how to combine or extend different existing matching approaches. Thereby, more comprohensive approaches (R2) can be achieved.

Aggregation of different matching results: Combining multiple different matching approaches or matching conditions within one approach leads to the challenge of how or whether to aggregate the results. Aggregation again forms a trade-off: On the one hand, aggregation conceales important characteristics of a matching result, on the other hand, it abstracts from a maybe large set of details and thereby simplifies further processing or keeps the overview for a human user. Thus, the decision has to be well elaborated depending on the context, e.g., the use case the matching is part of or the characteristics of the service descriptions.

Expressiveness of matching results: Just returning gradual results is not sufficient because of three reasons:

- Gradual matching results can be interpreted in different ways, depending on their origin: on the one hand they could reflect the amount of successfully matched conditions, on the other hand, they could also reveal the certainty of the checked conditions, e.g., due to heuristics used for evaluating the matching conditions. For example, there is a difference between: a) "a service matches 50%" and b) "we are 50% sure that a service matches". However, up to now, final matching results most often only reflect to which extent services match by refering to the amount of the matched conditions (as in a). However, since heuristics and simplifications are often required because of the trade-off between the reliability of matching results and the efficiency of a matching algorithm, results should also reflect how reliable a matching is (as in b).
- Having these different possibilities of interpreting gradual results, for each matching approach it has to be clearly defined, how its results are to be interpreted in order to be comprehensible.
- For further processing, it is not only relevant to which extent a service matches, but rather which conditions exactly did not match. Giving additional information about what is missing for a full match supports further processing, like service composition or adapter generation.

These points are not only challenges for future service matching approaches but also newly identified requirements for matching in OTF Computing. However, there is also a downside: When only exact matches are searched, it is possible to refuse a service as soon as one matching condition is not fulfilled, which allows to abort the whole matching process thereby to save time in many cases. However, if gradual results are desired, this is not possible. Even if the first condition has failed, all following conditions could be matched successfully and the service could still achive a high degree. Thus, the requirement for expressive matching results comes with the disadvantage of impeding the efficiency of the matching and sophisticated solutions to cope with this trade-off have to be found.

Dealing with incomplete knowledge: Up to now, there are only few approaches that explicitly deal with incomplete knowledge, although, as we already noted, it is most likely that service descriptions are not complete and important information that would be needed to determine an exact match are missing. Especially in uncertain environments like the semantic web or dynamic service markets in OTF Computing, we can assume that this challenge becomes increasingly relevant. Thus, research in this area is needed in order to get further insights about how to cope with it.

Explicit specification of variations: In order to find the best match that satisfies the requester's requirements, the requester should be able to explicitly specify which requirements are mandatory and which not. Especially the functional aspects lack of support for such explicit fuzziness in specifications. In QoS matching, few approaches support requester preferences to some extent, but most just deliver some ranking either assuming that the service with the most successfully matched conditions or that the service with the "best" (highest or lowest) quality values satisfies the requester's preferences best.

Furthermore, the differences in the terminology between the different approaches are notable, especially regarding fuzziness: There are many terms used similarly, like "fuzzy", "vague", or "imprecise" (see our website [45] for the complete list of terms we collected during the work on our survey). From this, we conclude that it is most important to clearly define the used "fuzzy" terms in future approaches.

6. RELATED WORK

Until now, there are no surveys or classifications for general service matching approaches. However, there are classifications of related matching approaches: semantic web service matchmakers and schema matching. Furthermore, there are also some partially related publications: a classification of component interoperability erros and surveys about web service composition and discovery. These publications show some similarities to our work but none of them explicitly evaluates the occurrence of fuzziness in service matching. In the following, we discuss the differences in more detail.

Related Matching Classifications.

In contrast to our classification, classifications of semantic web service (SWS) matchmakers are specialized for a special type of the matching approaches, namely approaches that use ontologies to match web services. The newest and most extensive classification has been published by Dong et al. [18] and enhances an earlier SWS classification presented by Klusch [31]. Similar to our classification, Dong et al. evaluate which service information is matched. However, they use the service specification languages the matching approaches are based on (e.g., OWL-S or SAWSDL) as a first-level classification property and put a strong focus on discovery mechanisms and architectures (e.g., centralized vs. decentralized). For our research goals, technical details are of less interest. Instead, we focus on the conceptual side, i.e., the contents to be matched and on the occurrence of fuzziness. Fuzziness is not explicitly considered in [18] or [31], only the approaches' matching degrees are presented.

Similarly, Bellur et al. [7] present a classification of SWS matchmaker algorithms, not regarding specific matching approaches but the underlying algorithms, e.g., greedy or description logic. In contrast to our paper, they only distiguish between two kinds of information that can be matched: input/output and precondition/effect. The only comparison criteria related to fuzzy matching they used are "ranking of hits" and "soft constraints". For our research, this is not sufficient because we are interested in more details regarding how fuzziness is caused and how it can be leveraged in order to improve concrete service matching.

Shvaiko and Euzenat [48] classify schema matching approaches. Schema matching approaches are only comparable to service matching to some extent because unlike services specifications, schemas typically only define static structural information which makes the task less complex. Nevertheless, there are some similarities to our work: The classification contains three different matching dimensions: input dimensions, process dimensions, and output dimensions. Since a service matching algorithm takes a service specification as an input, their input dimensions can be compared to our view-based classification. One of the output dimensions is concerned with whether the matching approach delivers gradual results, which is a criterion in our classification, too. The classification of process dimensions distinguishes between approximate and exact computations, similar to our fuzziness-based classification, but it does not take into account their sources.

Classification of Interoperability Errors.

Becker et al. present a classification of software component interoperability errors, i.e., interface mismatches [5]. They introduce different interface level classifications, e.g., an approach proposed by Beugnard et al. [9], in which interface models are classified in four levels: syntactic, behavioral, synchronization, and QoS. The most comprehensive classification Becker et al. present is derived from the Unified Specification of Components (UnSCom) framework [42]. These approaches are a good foundation for classifying service matching approaches. However, in order to answer research questions wrt. fuzziness, such approaches need to be combined with further criteria.

Web Service Composition and Discovery Surveys.

There are several surveys on web service composition and web service discovery. Since we consider service matching as an essential task to be accomplished during both service composition and service discovery, these publications are partially related to our work. However, only some of these surveys explicitly mention service matching. For example, Dustdar and Schreiner [19] classify service composition approaches and summarize that many approaches are based on ontologies and on ontological matching. Küster et al. classify "issues and approaches in automatic service composition" [34]. In their work, service matching is only a marginal issue, but they group some composition algorithms into different ways of "service chaining", a technique to compose services in chains in a way that their inputs and outputs or their pre- and postconditions match. How these matches are determined is not considered. Rambold et al. [47] survey different service discovery approaches. Regarding service matching, they distinguish between syntactical, semantical, and hybrid approaches. None of these surveys examines fuzziness.

7. CONCLUSIONS

In this paper, we performed a systematic literature survey of fuzzy service matching approaches. Driven by three research questions, we selected a set of approaches based on the requirements of OTF Computing, proposed a classification, and discussed the results. In particular, we derived three different sources of fuzziness and analysed how they are reflected in matching results (Q1). Furthermore, we discovered that there are no service matching approaches that satisfy all requirements from On-The-Fly Computing (Q2), as there is no comprehensive approach that supports matching of all five relevant aspects. Especially pre- and postconditions, protocols, and QoS properties are rarely matched in combination with each other (Q3). In addition, we identified future research challenges (Q3), e.g., the combination of different matching approaches and the generation of more expressive matching results. According to this, our classification successfully answers our research questions and provides valuable results for future research in the area of service matching. Thus, also paradigms similar to OTF Computing, like service-oriented computing and situational or pervasive computing benefit from our results as they also require adequate service matching approaches.

However, as already noted, the selection criteria for analysed matching approaches had to be strict and we are aware of the fact that some excluded publications might contribute to our research. Nevertheless, based on the lessons learned from our current survey results, we enable the development of an advanced classification in the future, as we are now able to distinguish between different forms of fuzzy matching and their consequences more precisely. Such an advanced classification could be based on improved search criteria and take into account more approaches. We are especially interested in earlier approaches from software component matching in order to learn from a broader area of research.

8. ACKNOWLEDGMENTS

The authors would like to thank Vanessa Petrausch for her help in creating the matching approaches classification and Sebastian Lehrig for reviewing the paper.

9. REFERENCES

[1] A. Algergawy, R. Nayak, N. Siegmund, V. Köppen, and G. Saake. Combining schema and level-based matching for web service discovery. Springer, 2010.

[2] D. Bacciu, M. Buscemi, and L. Mkrtchyan. Adaptive Fuzzy-valued Service Selection. pages 2467–2471. ACM, 2010.

[3] L. Bai and M. Liu. A Fuzzy-set based Semantic Similarity Matching Algorithm for Web Service. In *5th Int. Conf. on Services Computing (SCC)*. IEEE, 2008.

[4] L. Bai and M. Liu. Fuzzy sets and similarity relations for semantic web service matching. *Computers & Mathematics with Applications*, 61(8):2281–2286, 2011.

[5] S. Becker, S. Overhage, and R. H. Reussner. Classifying Software Component Interoperability Errors to Support Component Adaption. In *7th Int. Symposium on Component-Based Software Engineering (CBSE)*, Lecture Notes in Computer Science, pages 68–83. Springer.

[6] U. Bellur and H. Vadodaria. On Extending Semantic Matchmaking to Include Preconditions and Effects. In *15th Int. Conf. on Web Services (ICWS)*. IEEE, 2008.

[7] U. Bellur, H. Vadodaria, and A. Gupta. Semantic matchmaking algorithms. *Greedy Algorithms*, page 586, 2008.

[8] A. B. A. Bener, V. Ozadali, and E. S. Ilhan. Semantic matchmaker with precondition and effect matching using SWRL. *Expert Systems with Applications*, 2009.

[9] A. Beugnard, J. Jézéquel, N. Plouzeau, and D. Watkins. Making components contract aware. *Computer*, 1999.

[10] D. Bianchini, V. D. Antonellis, and M. Melchiori. Flexible Semantic-Based Service Matchmaking and Discovery. *17th Int. Conf. on World Wide Web (WWW)*, 2008.

[11] A. Brogi, S. Corfini, and R. Popescu. Semantics-based composition-oriented discovery of Web services. *ACM Transactions on Internet Technology (TOIT)*, 2008.

[12] C. Canal, P. Poizat, and G. Salaun. Model-based adaptation of behavioral mismatching components. *IEEE Transactions on Software Engineering*, 2008.

[13] Y. Chabeb, S. Tata, and A. Ozanne. YASA-M: A Semantic Web Service Matchmaker. IEEE, 2010.

[14] J. C. Corrales, D. Grigori, M. Bouzeghoub, and J. E. Burbano. BeMatch: A Platform for Matchmaking Service Behavior Models. ACM, 2008.

[15] A. Cuzzocrea and M. Fisichella. Discovering semantic Web services via advanced graph-based matching. In *Int. Conf. on Systems, Man, and Cybernetics (SMC)*. IEEE, 2011.

[16] D. A. D'Mello, V. S. Ananthanarayana, and T. Santhi. A QoS Broker Based Architecture for Dynamic Web Service selection. IEEE, 2008.

[17] D. A. D'Mello, I. Kaur, N. Ram, and V. S. Ananthanarayana. Semantic Web Service Selection Based on Business Offering. IEEE, 2008.

[18] H. Dong, F. Hussain, and E. Chang. Semantic Web Service matchmakers: state of the art and challenges. *Concurrency and Computation: Practice and Experience*, 2012.

[19] S. Dustdar and W. Schreiner. A survey on web services composition. *Int. Journal of Web and Grid Services*, 1(1):1–30, 2005.

[20] G. Fenza, V. Loia, and S. Senatore. A hybrid approach to semantic web services matchmaking. *Int. Journal of Approximate Reasoning*, 48:808–828, 2008.

[21] J. L. Fiadeiro, A. Lopes, and L. Bocchi. An abstract model of service discovery and binding. *Formal Aspects of Computing*, 23(4):433–463, 2010.

[22] F. Giunchiglia, F. McNeill, M. Yatskevich, J. Pane, P. Besana, and P. Shvaiko. Approximate structure-preserving semantic matching. *On the Move to Meaningful Internet Systems*, 2008.

[23] D. Grigori, J. Corrales, and M. Bouzeghoub. Behavioral matchmaking for service retrieval: Application to conversation protocols. *Information Systems*, 2008.

[24] D. Grigori, J. Corrales, M. Bouzeghoub, and A. Gater. Ranking BPEL Processes for Service Discovery. *Transactions on Services Computing*, 2010.

[25] Y. Hao, Y. Zhang, and J. Cao. Web services discovery and rank: An information retrieval approach. *Future Generation Computer Systems*, 26(8):1053–1062, 2010.

[26] Z. HongKang, Y. XueLi, Z. GuangPing, and H. Kun. Research on Services Matching and Ranking Based on Fuzzy QoS Ontology. In *Int. Conf. on Computational Aspects of Social Networks*. IEEE, 2010.

[27] B. Jeong, H. Cho, and C. Lee. On the functional quality of service (FQoS) to discover and compose

interoperable web services. *Expert Systems with Applications*, 36(3):5411–5418, Apr. 2009.

[28] C. Ke and Z. Huang. Self-adaptive Semantic Web Service Matching Method. *Knowledge-Based Systems*, 2012.

[29] B. A. Kitchenham and S. Charters. Guidelines for performing Systematic Literature Reviews in Software Engineering. 2007.

[30] A. Klein, F. Ishikawa, and S. Honiden. Efficient qos-aware service composition with a probabilistic service selection policy. In *8th Int. Conf. on Service-Oriented Computing*. Springer, 2010.

[31] M. Klusch. Semantic web service coordination. In *CASCOM: Intelligent Service Coordination in the Semantic Web*, pages 59–104. Springer, 2008.

[32] M. Klusch and P. Kapahnke. iSeM: Approximated Reasoning for Adaptive Hybrid Selection of Semantic Services. *The Semantic Web: Research and Applications*, pages 30–44, 2010.

[33] M. Klusch and P. Kapahnke. The iSeM Matchmaker: A Flexible Approach for Adaptive Hybrid Semantic Service Selection. *Journal of Web Semantics: Science, Services and Agents on the World Wide Web*, 2012.

[34] U. Küster, M. Stern, and B. König-Ries. A classification of issues and approaches in automatic service composition. *Int. Workshop on Engineering Service Compositions (WESC)*, 5, 2005.

[35] Q. A. Liang and H. Lam. Web Service Matching by Ontology Instance Categorization. IEEE, 2008.

[36] W. Lin, C. Lo, K. Chao, and M. Younas. Consumer-Centric QoS-aware Selection of Web Services. *Journal of Computer and System Sciences*, 2008.

[37] F. Liu, Y. Shi, J. Yu, T. Wang, and J. Wu. Measuring Similarity of Web Services Based on WSDL. In *IEEE 17th Int. Conf. on Web Services (ICWS)*. IEEE, 2010.

[38] M. Liu, W. Shen, Q. Hao, and J. Yan. An weighted ontology-based semantic similarity algorithm for web service. *Expert Systems with Applications*, 2009.

[39] M. Liu, W. Shen, Q. Hao, J. Yan, and L. Bai. A fuzzy matchmaking approach for Semantic Web Services with application to collaborative material selection. *Computers in Industry*, 63(3):193–209, Apr. 2012.

[40] T. Mens. On the Complexity of Software Systems. *IEEE Computer*, 45(8):79–81, 2012.

[41] M. Naeem, R. Heckel, F. Orejas, and F. Hermann. Incremental Service Composition Based on Partial Matching of Visual Contracts. In *13th Int. Conf. on Fundamental Approaches to Software Engineering (FASE)*. Springer, 2010.

[42] S. Overhage. UnSCom: A Standardized Framework for the Specification of Software Components. *Object-Oriented and Internet-Based Technologies*, pages 291–307, 2004.

[43] M. Palmonari, M. Comerio, and F. D. Paoli. Effective and Flexible NFP-Based Ranking of Web Services. Springer, 2009.

[44] Z. Peng, W. He, and D. Chen. Research on Fuzzy Matching Model for Semantic Web Services. IEEE, 2008.

[45] M. C. Platenius, M. von Detten, S. Becker, W. Schäfer, and G. Engels. A Survey of Fuzzy Service Matching Approaches in the Context of On-The-Fly Computing - Website. http://goo.gl/uJhWj, Last Access: April, 2013.

[46] P. Plebani and B. Pernici. URBE: Web Service Retrieval Based on Similarity Evaluation. *IEEE Transactions on Knowledge and Data Engineering*, 21(11):1629–1642, 2009.

[47] M. Rambold, H. Kasinger, F. Lautenbacher, and B. Bauer. Towards Autonomic Service Discovery Ű A Survey and Comparison. IEEE, 2009.

[48] P. Shvaiko and J. Euzenat. A Survey of Schema-based Matching Approaches. *Journal on Data Semantics IV*, 3730:146–171, 2005.

[49] D. Skoutas, D. Sacharidis, V. Kantere, and T. Sellis. Efficient Semantic Web Service Discovery in Centralized and P2P Environments. In *7th Int. Conf. on The Semantic Web (ISWC)*, pages 583–598. Springer, 2008.

[50] D. Skoutas, D. Sacharidis, A. Simitsis, V. Kantere, and T. Sellis. Top-k Dominant Web Services Under Multi-Criteria Matching. In *Int. Conf. on extending database technology: advances in database technology*. ACM, 2009.

[51] D. Skoutas, D. Sacharidis, A. Simitsis, and T. Sellis. Serving the Sky: Discovering and Selecting Semantic Web Services through Dynamic Skyline Queries. IEEE, 2008.

[52] G. Spanoudakis and A. Zisman. Discovering Services during Service-based System Design using UML. *Transactions on Software Engineering*, 2010.

[53] V. X. Tran, S. Puntheeranurak, and H. Tsuji. A new service matching definition and algorithm with SAWSDL. *3rd Int. Conf. on Digital Ecosystems and Technologies*, 2009.

[54] University of Paderborn. Collaborative Research Center “On-The-Fly Computing” (CRC 901). http://sfb901.uni-paderborn.de, Last Access: February 21, 2013.

[55] P. Wang. QoS-aware web services selection with intuitionistic fuzzy set under consumer's vague perception. *Expert Systems with Applications*, 2009.

[56] F. You, Q. Hu, Y. Yao, G. Xu, and M. Fang. Study on Web Service Matching and Composition Based on Ontology. *WRI World Congress on Computer Science and Information Engineering*, pages 542–546, 2009.

[57] B. Yun. A New Framework for Web Service Discovery Based on Behavior. *5th IEEE Asia-Pacific Services Computing Conf. (APSCC)*, 2010.

[58] K. Zamanifar, G. Heidary, N. Nematbakhsh, and F. Mardukhi. A new approach for semantic web matching. *Security-Enriched Urban Computing and Smart Grid*, pages 77–85, 2010.

[59] A. Zaremski and J. Wing. Specification Matching of Software Components. *ACM Transactions on Software Engineering and Methodology (TOSEM)*, 1997.

[60] Y. Zhang, Z. Zheng, and M. Lyu. WSExpress: A QoS-Aware Search Engine for Web Services. In *IEEE 17th Int. Conf. on Web Services (ICWS)*. IEEE, 2010.