PERSONALIZED WEB SERVICE SELECTION

Shailesh Khapre and D. Chandramohan
Department of Computer Science, Pondicherry University, Pondicherry, India
shaileshkhaprerkl@gmail.com, chandrumeister@gmail.com

ABSTRACT

Web services have shown the potential to have a significant impact on the business, however, also shown problems like automatic service discovery or service interoperability issue. Because of today's wide variety of services offered to perform a specific task, it is essential that users are guided in the eventual selection of appropriate services. In this paper we present an extensive study of different selection techniques towards advanced personalization of Web service selection. We propose the customization of the user profile to support different steps of interaction with services and propose techniques to personalize each subsequent step. Our main contribution is an algorithm featuring the expansion of service requests by user-defined demands and wishes. Services not matching a user specification are discarded on the fly and equally useful results of alternative services can be compared with respect to user provided preferences. We also present a case study to evaluate our personalization techniques.

KEYWORDS

Web Services Selection, Registries and Semantic Discovery, Personalization, Collaborative Filtering, Service/ Usage Patterns

1. INTRODUCTION

Web services are Internet-based, distributed self-contained modular applications that provide standard interfaces and communication protocols aiming at efficient and effective service integration. They have showed their usefulness in a wide variety of domains. Typical Web service applications include business-to-business integration (B2B), business process integration and management, e-sourcing and content distribution. Web service interfaces and bindings are defined, described and discovered by XML parsers, supporting direct XML message-based interactions with other services and applications via Internet-based protocols like SOAP [30]. Standards for service description and discovery such as the Universal Description Discovery and Integration (UDDI) [33] specification are designed to function in a fashion similar to white pages or yellow pages, where businesses and services can be looked up by name and/or by standard service taxonomy. UDDI provides a framework for the description of basic business and service information and offers a simple, yet extensible mechanism to provide detailed service access information on the basis of the Web Services Description Language (WSDL) [8].

Initially Web services were mainly intended to engage in dynamic business-to-business (B2B) interactions with services deployed on behalf of other enterprises or business entities. However, with the evolution of Web service technology networked services will not only become increasingly sophisticated, but also move into the area of business-to-consumer (B2C) or even peer-to-peer interactions (P2P). In [35] we have already pointed out that user interaction with Web services will mostly be on a multi-Interaction per-service. Instead, the ultimate goal in personalized service provisioning has to be the fulfilment of specific user needs expressed as complex tasks which are normally further divided into simpler sub-goals and subsequently matched to different services. With the number and diversity of services expected to grow, adequate techniques for user-centric and user preference-based service discovery and selection will be needed. Even though UDDI and WSDL are commonly used today to publish an implemented service they still essentially lack strong concepts for service personalization which

are crucial for advanced usability. In this paper we try to address some of the most important challenges arising from the personalization of Web services. We study how basic service cataloguing can be enhanced with concepts from cooperative databases, [26] and collaborative filtering [22] to discover, select and combine services according to the special needs and preferences of an individual user. In this context, discovering service usage patterns and modelling patterns for typical user needs together with the explicitly or implicitly given preferences of single users or user groups can be used for an efficient and effective Web service matchmaking.

The rest of this paper is organized as follows: In the following section we have outline our ideas towards an advanced selection of Web services and present architecture for the personalized discovery and selection. In section 3 we briefly survey the existing & related state of the art in Web service cataloguing and publishing and discuss related work concerned with the enrichment of standard lookup services. Section 4 illustrates the use of fundamental concepts from cooperative service databases and collaborative filtering: cooperative answering and user preference-based modelling. We present a basic algorithm for personalised web service selection and composition in detail together with a sample usage scenario in section 5 and close with a summary and brief outlook on our future work.

2. PERSONAL SERVICE SELECTION

Today's Web service platforms are beginning to offer technology that already supports simple forms of personalization. For instance, Microsoft's .NET Passport authentication service [23] enables client to enter and store simple profile information (e.g., e-mail address and password, mailing address, etc.) to enable client to sign-on once with a member of an affiliated group of organizations and subsequently use various sites offered by other group members without the need for signing-on times and again. In addition, services like e.g. the .NET wallet service will store credit card information so those clients do not need to re-enter it for every transaction, either. In parallel, the Liberty Alliance Project [31] was formed as an organization of major information technology and telecommunication companies that deliver's and support's a similar federated network identity. This strongly emphasizes the importance that is assigned to personalization as a basic necessity for successful service delivering.

However, although technologies such as the Liberty Alliance or .NET passport herald an essential paradigm shift in Web service deployment from mere B2B automation of business processes to more consumer-oriented network services, up to now they provide virtually no advanced personalization concepts. Especially the discovery and/or selection of services regarding the personal preference/requirements of a user are not yet considered. In the following we consider the essential steps in selecting satisfactory Web services for individual users.

2.1. User-centered Web Services

Initially Web services were mainly intended to engage in dynamic business-to-business (B2B) interactions with services deployed on behalf of other enterprises or business entities. Broad interest in standardization/customization efforts was aimed at reducing the necessary user interaction. However, with the advancement of Web service technology the complexity of possible tasks and the availability of services anytime anywhere, e.g. through powerful mobile client devices, will strongly increase. Thus networked services will not only become increasingly sophisticated, but also move into the area of business-to-consumer (B2C) interactions bringing back the user as active participant during interactions. The 'user in the loop' will enhance the service offerings and increase the competing variety that can be sensibly provisioned, but due to this excess information will also demand a great deal of personalization. The development of Internet portals like Yahoo.com already demonstrates the usefulness of a user-centred approach for e.g. social-networking and related areas. But also here the necessity to

provide a personalization/customization mechanisms or a support for browsing available services has become apparent to deal with the number of different offerings that may be more or less relevant for each individual user. For instance, MyYahoo.com allows individual users to define their own portal entry page and determine which personal content to display or which services to activate on start-up. With Web services becoming an integral component of the future Web, a user-centred approach to personal Web services is easily conceivable, e.g. through Web service technology in the backend of Internet portals. In the following we will deal with Web services in the context of being user-centred Services offered via the Internet.

The problem of how to interactively deal with Web services of course will have some impact on the areas of human computer interaction (HCI) and user interface design (UID) (e.g. for mobile devices with limited capabilities) like it has already been the focus of research activities in setting up traditional portal pages. But since dealing with these issues is not the focus of the work presented here, we will assume the existing graphical user interfaces also as a basis for our necessary customer-service (C2S) interaction. Trying to model the usage of Web services along the lines of traditional portal pages, but using the services full capabilities and the powerful technologies, interaction with a truly user-centred service can thus be divided into three major phases (see fig. 2):

- the discovery of services that are basically able to perform a task based on a user's service request,
- the querying of services for subsequent selection based on user preferences for deliverable objects,
- the final execution of a service, after a decision for one of the available objects.

Of course not every Internet service that is deployed as a Web service will be taken into account as user-centred and needs personalization. We basically distinguish between simple services that can always be executed and just return a simple result, e.g. a currency converter or a weather forecast, and complex services, e.g. flight booking, restaurant reservation or a travel agency. Unlike for a currency conversion service that converts say US Dollars to Japanese Yen, for a flight booking service it is not obvious that the execution of the service will deliver the likely result and fulfil a user's needs. Its sensible execution may be dependent on some key information, e.g. whether a flight with a certain airline can be booked on a certain date. Portals like Expedia.com [15] therefore pose the user’s request to a variety of hard-wired services and display the collected results to the user. However, the hard-wiring of the Expedia portal does not allow for the flexibility of the Web services paradigm and thus does not really meet the challenges of the Internet. We thus have to consider the questions, if a service generally has the capability of accomplishing given task and if the desired result is always available, if a certain service is executed. Of course, if we need a service like currency conversion that always is able to deliver the expected result, the process of selecting an adequate service for execution is rather clear-cut, e.g. depending on the execution costs, or guaranteed accuracy, etc. The selection process will, however, be a lot harder in the case of e.g. flight booking where the quality of deliverable results -in this case certain flight offerings- have to be compared. We will deal in detail with this problem in section 4.

Web services thus can be divided into business-to-business (B2B) and business-to-consumer (B2C) services and we can further subdivide the latter ones into always applicable services and those whose applicability in each specific case still depends on a specific piece of information. From a deliverer point of view the last class of services poses the most problems with respect to personalization, because:

- They usually model complex user-centred tasks consisting of different components like advertising, information and execution
- The final decision which service to execute almost always has to be performed directly by the user, because different services may offer various advantages past the specified user preferences.

2.2. Selection Steps

Let us now deal with the several steps that are necessary in selecting satisfactory services for subsequent execution. Again we will focus on more advanced services, where the general capability and the availability of suitable results have to be considered. A successful user interaction with a service will generally consist of a complex goal statement followed by different discovery/interaction steps resulting in an finest service selection. In detail we distinguish:

The user's intentional goal: The purpose of user interacting with a Web service is always a certain task a user wants to perform. The definition of this goal can be quite complex depending on the user's demand. There may be single time goals like ordering a item or arranging for a business trip. But there can also be permanent goals like managing communication tasks according to a user's current device or condition.

The discovery of available services: The service discovery will show user which services can be possibly used to perform a certain task. It depends on the reachability of services and the device that a user plans to perform the service with. Sometimes services are offered for a variety of devices automatically adapting the content concerned. The discovery is on the whole is an automatic step performed by the user's device.

The service composition: For complex goals there may be no adequate services discovered that perform the entire task, but only parts of it. The decomposition of a complex task into sub-goals which can be performed by simple services is a tricky matter. It strongly depends on the user choose strategies to fulfil the entire task.

The service selection: The service selection is the step of deciding which service(s) to use to finally perform the overall task. For instance there may be services performing similar goals and users can select the one that is nearby to his/her intention or there may be different services performing the same goal, however offering different objects, at different costs, quality levels, etc.

We will revisit all these steps in the course of our paper and mention possibilities to personalize each step. As can be easily seen, personalization will play a key role in the discovery, selection and composition of services: whereas the automatic discovery of unnecessary or unsuitable services is just a matter of inefficiency, the flooding of users with irrelevant services utility in the selection process can cause severe problems in usability. What is more, the refined adaptation of decomposition strategies to the specific user's notion of will result in higher satisfaction and thus in higher quality of service.

2.3. A Usage Scenario

Consider a sample user called Michael, who is working in a California Company and has to attend a business meeting in Boston during the next week. Setting up all necessary preparations is a complex matter and finding satisfactory services can already be quite time-consuming. But communicating personal requirements and preferences to many services that even might not be suitable for Michael's specific task will definitely get tiresome. Basically the task may include, but is not restricted to setting up:

- The necessary transportation
- Reservations for accommodation
- Arranging meeting times and locations with business partners

Depending on Michael's preferences it may also involve other tasks as for instance arranging entertainment for the evenings or discovering sporting or sightseeing possibilities.

Throughout this paper we will consider Michael's typical tasks to show how different personalization schemes/techniques can be applied to improve the overall quality of service. Figure 1 shows how we envision the collaborative support of personalized service discovery and selection for our test user Michael: existing UDDI-based service registries' are integrated into a Meta repository enabled for cooperative search. Beside the integrated Web Service registries the Meta repository also hold service usage patterns associated with the typical usage of certain services by particular users or user groups.

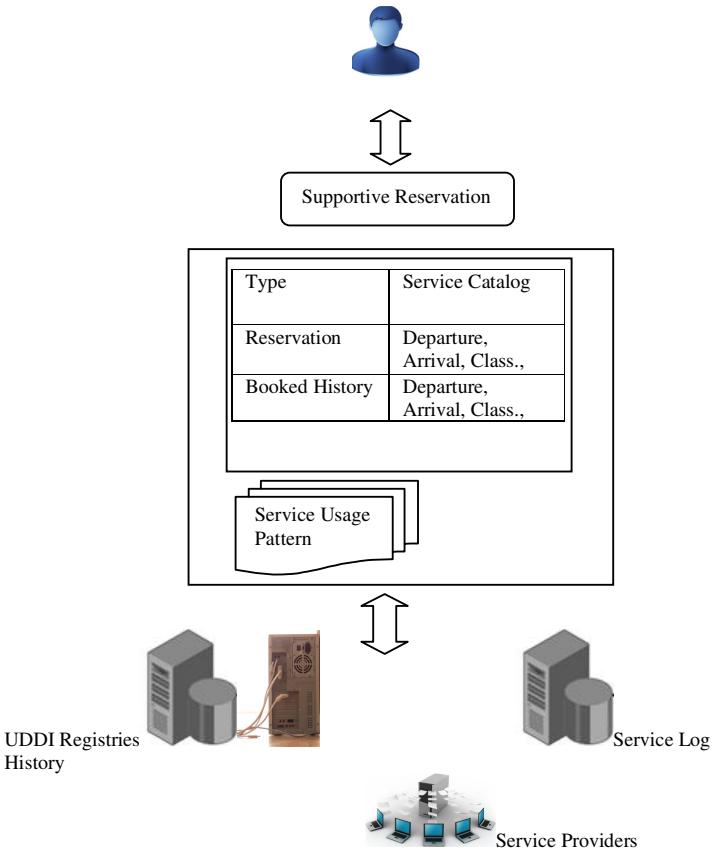

Figure 1. Personalized Service Discovery and Selection

3. STATE OF THE ART AND RELATED WORK

With indication to our loom to bespoke examination detection and collections, temporary investigations are departure on Web service consistency performance communicate to other job apprehensive with superior service sketch and commercial survival.

UDDI (Universal Description, Discovery and Integration) [33] is an enterprise registry wished by Microsoft, IBM, Amazon and Ariba to widen a paradigm for an online registry of Web services. UDDI assist to circulate and energetic innovation of arrangement services and allocates developers to situate services for undeviating incantation or assimilation into new multifaceted services. Web service contributors record its commercial pack beside with keywords for labelling. Prospective service users will regain industrial pack of the registry foundation on keyword search. It is understood that abuser and contributor use the similar deposit of keywords for service depiction. The search apparatus relies on pre-defined keyword labelling and does not transfer to the semantic substance of the poster.

WSDL (Web Services Description Language) [7] is an XML glossary, strongly coupled with UDDI as the layout for identifying the boundary to Web services registered with a UDDI repository. WSDL endeavour to detach services from the tangible records format and procedure used for working. It consequently illustrates a required format between the conceptual service explanation and its explicit performance. Communication between generalizations of services is at a relatively low level in terms of thought from significance code of behaviour, service bindings and announcement harbour. WSDL has a perception of participation and productivity types but, like UDDI, does not maintain semantic explanation of services.

E-Speak [13] have an enterprise driven by Hewlett-Packard to enable enhanced service discovery. E-speak and UDDI have similar goals in that they both facilitate the advertisement and discovery of services. E-speak is comparable to WSDL in that it supports the description of service and data types and characteristics a corresponding service that balance service requests with service metaphors. This is prepared largely on the basis of input-output and service type harmonizing. E-speak illustrate services as a set of characteristic within numerous dissimilar terminologies which are deposit of characteristic universal to a coherent grouping of services. Research requirements are coordinated beside service metaphors with esteem to these distinguishing. Presently, there is no semantic significance attach to any of the characteristic. Any consequent which takes position to prepare over the service metaphors keywords does not discriminate between any additional subtypes.

DAML-S [1] is an ontology-based evolution to the portrayal of Web services residential as part of the DARPA instrument mark-up idiom program. DAML-S aims at as long as a universal ontology of services and stimulated by previous research in the area of so-called semantic Web that includes pains to colonize the Web with contented and services having recognized semantics. Build on top of DAML+OIL [10], the design of DAML-S pursues the encrusted progress to semantic Web markup languages. The ultimate goal of DAML-S is to present ontology that authorize transmit criterion and abuser to discern, raise and arrange Web services. Currently the structure of the DAML-S ontology is threefold and consists of a service profile for promoting and discovering services, a procedure model which gives a comprehensive explanation of a service's procedure and a service instruction which afford trivia on how to interoperate with a service via communication substitute.

ebXML [14] is primarily developed by OASIS and the United Nations. It approaches the behaviour of service metaphors from a workflow perception. ebXML uses two views to illustrate business exchanges, a Business Functioning View (BFV) and a functional service view (FSV). In ebXML the BOV deals with the semantics of business data transactions between Web service providers. The FSV deals with the opinionated services themselves, i.e. their competence, boundary and procedure. It has the perception of association procedure sketch which allows a traffic collaborator to articulate their sustain company progression and business

service edge requests by other ebXML accommodating operate collaborator. A production progression is a set of production manuscript interactions between traffic colleagues. It restrains industry cataloguing, contact in sequence, maintain business progression, interface requests etc. They are catalogue within an ebXML registry, which assist the discovery of other employment colleague and the business development they support. In this respect, ebXML has some correspondence with UDDI.

WSMF (Web Service Modelling Framework) [16] afford an additional perception foundation on semantic web technology for embryonic and recounting web services and their composition. WSMF illustrate the qualification and post-condition of services together with a service representation. WSMF aims at sturdily de-coupling the combination of apparatus that put into action a web service application while at the same time provided that a maximal quantity of negotiation connecting the different apparatus. WSFM builds a complete ontology such as DAML-S and make available notion of aspiration repositories and intermediaries to solve composite service desires.

4. COOPERATIVE SERVICE SELECTION

The concepts and standards outlined above enable the basic discovery steps of Web services. However, after discovery Web service developers and individual still users have to select best from one or a composition of services that best serve their specific task. Given the growing diversity of services users can't be expected to browse all service offers until they find an satisfactory match. Thus, the task of service selection and composition has to be supported by leading to a recommendation of a set of possible services or combinations between which users can subsequently choose the best.

So far we have only spoken about personalization techniques to find or compose the suitable service to suit a specific user's goal. But how does a user know from the service description, if the specific object he or she wants to access or deal with is offered? Assume that a user has discovered a service for booking flights. Stating that she/he needs a flight booking service may not be enough - if a user is not sure that the specific flight on a specific date he needs can be booked by the service. Hence, after services have been chosen, a user's profile has to perform another important filtering task. While using the service, it has to supply user preferences with respect to the values for parameters that are transferred to the service.

Thus Preferences and Profiles are needed for:

- **Decomposing goals**: preferred strategies, disliked decompositions, etc.
- **Choosing services**: execution cost, accuracy, trustworthiness, etc.
- **Supplying service constraints**: preferred input values, hard or soft constraints, ordering of relaxation, etc.

Expanding the query for a service with these preferences can essentially improve the quality of selection by ruling out unsatisfying services. However, by adding too many query terms a search might easily become too definite and return empty results. Thus for the personalized selection of services based on input constraints we need a cooperative service behaviour that can also handle soft constraints. Cooperative retrieval techniques have been introduced by the research area of cooperative answering [26] based on the experience that finding the exact query to pose against a database is a difficult matter. In contrast to traditional databases - which use a perfect match retrieval model - any constraint in the query can be fulfilled or is violated by the objects in the database. The result set contains all those objects that fulfil every constraint of the query, i.e. all exact matches. Thus a too specific query will return no results at all, while a too general query floods the user with result objects.

However, at the time of framing a query a user can't be sure about the content of the database. Hence a typical behaviour in querying databases is starting with a more general query and getting more specific, if too many results are returned or starting with a quite specific query an simplifying it by dropping attributes, if no results are returned. In traditional SQL databases this task has to be performed by the user. Enabling a user to decide which parts of a query should be kept hard-wired and which parts (and in what order) can be gradually relaxed is a helpful technique. If a user for instance didn't specify an airline in his query to a flight booking service, the choice of transportation can, according to this query, be relaxed to any value. Thus - as shown above - the possible booking services can be generalized to all those that have or don't have a carrier input constraints. Nevertheless, a user might have certain preferences for a precise airline. The semantics of dropping it out of the query means that a user does not insist on this point (in contrast to e.g. a specific arrival date or the destination of the flight). If many possible alternatives to choose from are returned, applying more user specific information can considerably reduce the answer set to the most desirable ones.

Those preferences can be gathered from previous interactions in form of a profile management or directly be specified by the user in the form of soft constraints. Unlike hard constraints those parts of the query can be relaxed, if the query gets too precise and does not return some suitable answers anymore. In our example from above we would add a preferred transportation attribute to our query (if the chosen service supplies such an attribute) and pose the query. Note that the expected result set will due to the more specific query generally contain fewer results avoiding the flooding effect. But since the additional attributes have been chosen as soft constraints, they can be relaxed if necessary, thus avoiding null result sets.

Cooperative answering techniques and their incorporation into database systems is an active field of research: the first cooperative systems [22] relaxed attributes step by step in a round robin fashion, if a query was too precise and did not return results. More user-centred approaches like [9] allowing users to specify the order and steps in which to relax each attribute lead to results of higher quality. Recently the management of complex preference patterns and the conversion into declarative database languages has gained more and more attention. Languages like PreferenceSQL [18] or the skyline-operator [6] already allow posing declarative database queries with soft constraints filtering irrelevant of result objects.

5. INTERACTING WITH WEB SERVICES

Discovering services using existing standard technology (cf. section 3) is mainly a task of keyword-based searches in more or less complex service descriptions. It is beyond the scope of this paper to comment on all possible problems that occur when trying to discover services for user specific tasks. Using today's techniques the user has for instance to use the precise phrases and ontology that is provided by standards like WDSL. Improvements for this situation are given by several existing techniques for matching of ontologies [12] or expansion by suitable thesauri [2]. In the following we will deal with the task of selecting appropriate services, after a discovery has yielded services matching the task a user is interested in.

5.1. Service Selection

In contrast to the task of composing Web services the selection of the right services strongly depends on the costs of a specific service or the objects or meta-data that a service can provide. For instance there may be a variety of services revealing the location of certain individuality, but at different costs or with different accuracy. The same applies for more complex service like e.g. flight booking, where a user has to know if the service offers a certain flight that satisfy his or her needs, before using the service. Therefore two aspects are necessary for service selection which is:

The service has to:

- be generally able to perform the desired task
- offer the specific object/information the user is interested in

Thus users first have to query all probable services that generally perform their task, if they will also satisfy their specific need. Assuming that a set of services has been discovered, we will now discuss those personalization techniques that lead to an eventual choice of services presenting the best quality for each specific user. We will first revisit our motivating example, then state the algorithm and perform a test interaction.

Assume that our user Michael has to book a flight from California for his business trip to Boston. Since the business meeting starts at 3 p.m. on the 10th of November, he has to arrive till noon to have some time left to get to the business location. Besides, he wants to fly business class. The SQL-like query:

```
SELECT flights FROM service
WHERE departure = 'CAL' AND arrival = 'BOS'
AND arrival_date <= '10-11-2002 12:00'
  AND class = 'business'
```

would express his needs. Please note that all test values specified in the where clause will be considered hard constraints. Neither would Michael at this stage will fly to New York instead of Boston, nor would he consider a flight that does not arrive in time to get to the meeting.

Our algorithm shows the necessary steps that have to be performed in order to choose adequate services and get the highest quality results. It assumes that a goal is provided that can be managed by Web services, a user profile knowledge and/or general common knowledge exists and that a common vocabulary is used. For convenience we will abstract from techniques for ontology matching, etc. that could be applied in cases with no shared vocabulary.

Algorithm: Selecting a Service

1. Perform a keyword-based search on the service descriptions to find services for the specified goal.
2. Group all discovered services by signature parameters and discard all services that do not support querying with all user-provided query terms (explicit hard constraints).
3. Get the service parameters that are offered beyond those covered by hard constraints (user-Preferences).
4. If existent, get preferred values for the parameters collected in step 3 from user profiles & domain knowledge. (Implicit soft constraints).
5. Expand the user's service request with these discovered soft constraints and query the services from step 2 with their respective signature.
6. Collect all services results and order them by their utility, e.g. assign highest utility to those results meeting the most constraints.
7. If no results are returned, invoke the user reconsider the hard constraints and start over with step 2.

Let us consider how this algorithm works continuing our example: Michael wishes to book a flight. First we have to discover all different services that allow performing the task of booking a flight and check, if flights meeting Michael's criteria are available (step 1). A keyword search on service descriptions might result in a list of four services for flight booking given in table 1.

Table 1. Discovered Services for Flight Booking

Service	Input Parameters	Semantic Description
Booking	Departure, Arrival, Class, Departure Date, Arrival date	Flight Booking
Flights On-Line	Departure, Arrival, Class, Departure Date, Arrival date, Airline	Flight Booking, Flight Information
Air Travel Economy	Departure, Arrival, Class, Departure Date, Arrival date, Airline	Economy class flight booking
Easy Flights	Departure, Arrival, Class, Departure Date, Arrival date, Airline, Price, non-stop	Flight Booking

The next step is sorting out what parameters are required to guarantee the observance of the hard constraints (step 2). Since our query specifies departure, arrival, arrival date and class, we have to discard the Air Travel Economy service, as class input parameter is provided. Step 3 checks for additional parameters past the hard constraints and gets departure date, airline, price and non-stop. Now in step 4 Michael's profile knowledge has to be queried for preferences with respect to these additional parameters and reveals that Michael generally uses to fly Delta Airlines and prefers non-stop flights. Some general preferences from the domain could also be applied like "everyone prefers a short travelling time" (i.e. a departure date with maximum proximity to the arrival date is preferred). However, since it is a business trip, we may get no information regarding preferred prices from Michael. Thus no parameter for the price will be given in the queries. Now we are ready to build the queries regarding the services (step 5). Using again the declarative notation of [18] we get three queries that can be posed to the services:

```
SELECT flights FROM Book'n_Fly
WHERE departure = 'LAX' AND arrival = 'BOS'
  AND arrival_date <= '10-11-2002 12:00'
  AND class = 'business'
PREFFERING MAX(departure_date)
```

```
SELECT flights FROM Flights_On-line
WHERE departure = 'LAX' AND arrival = 'BOS'
  AND arrival_date <= '10-11-2002 12:00'
  AND class = 'business'
PREFERRING MAX(departure_date)
       AND airline = 'Delta'
```

```
SELECT flights FROM Easy_Flights
WHERE departure = 'LAX' AND arrival = 'BOS'
  AND arrival_date <= '10-11-2002 12:00'
  AND class = 'business'
PREFERRING MAX(departure_date)
       AND non-stop = true
```

Table 2 shows sample results from our Web services. Please note that even though some services could not be queried on airlines all services may be capable of stating the airline in the result set. Easy Flight delivers two likely flights, because one is a non-stop flight whereas the other one has a better departure date. Since Michael didn't state an ordering those flights can be considered equally good.

Table 2. Sample Service Results

Service	Results			
	Name Airline	Departure Date	Arrival Date	Class/ Non-Stop
Bookings	UA999	LXX	BOS	Business
	United	10-11-2010 06:25	10-11-2010 11:40	---
Flights	BA99	LXX	BOS	Business
On-Line	British Airways	10-11-2010 05:25	10-11-2010 10:40	---
Easy	D765	LXX	BOS	Business
Flights	Delta	10-11-2010 04:15	10-11-2010 11:40	True
Easy Flights	LH1737	LAX	BOS	Business
	Luf-thansa	10-11-2010 06:35	10-11-2010 11:55	False

Performing step 6 of our algorithm we could order the result set suggesting Easy Flight's D765 as best result since it optimizes two soft constraints (airline and non-stop), whereas all other flights only meet one of the constraints. However, also every other suitable utility assessment can be used: Michael might for instance insist on the shortest travelling time. Returning a non-empty result set the selection is finished and our user can eventually decide which service should be used for the booking Flight. However, if no results had been returned, Michael would have had to reassess his constraint and might e.g. have decided to drop the business class constraint. Then we would have to start all over and please note that in this case even the previously discarded Air Travel Economy service would have to be reconsidered.

Our assumption that all user supplied query terms have to be hard constraints is only for our convenience. The generalization to a case in which a user can also explicitly provides soft constraints or a order of relaxation is straightforward. These soft constraints would have to be seen as supplied for this special user service request and could be more specific than or even conflicting with general long-term preferences and would thus have to be evaluated after the provided hard constraints, but before the query is expanded with terms from the long-term profile.

5.2. Service Composition

So far we have considered concepts for a simple service discovery and shown how to select single services in a personalized mode. But whereas the discovery of services can often be performed using simple keyword-based searches on the semantic service description, a combination of several services still poses complications. First there are technical problems dealing with the ways to summon services and the flow of data between different services. Due to the heterogeneity and autonomy of different services it is not possible to depend on the current models to build and coordinate compositions of web services. Complex interactions need high-level support like intermediary services or transaction models. These rather technical problems have already gained wide attention and languages like WSFL [20], XLANG [32] or WSCL [2] have defined primitives for composing services and automated service coordination. But recent work has also already considered some advanced interaction concepts. For instance [7] proposed dedicated combining services delivering a process model for each user's selection of services, [29] presented a XML-based transaction model and [27] tackled the logic-based automated validation and composition of web services in the area of semantic web.

On the other hand we have yet again personalization problems in choosing services for reasonable combinations. Consider for instance our sample user Michael having the target of

attending a business meeting in a different city. Depending on the characteristics of the starting point and the meeting place several ways of achieving the target can be applied possibly involving a variety of different services. If for instance the journey spans over a rather long distance a service for booking a flight will possibly be involved. However, Michael may also decide to travel by train. Since flights can only supply transportation between airports, the goal is decomposed into several pieces such as: finding possible or most suitable airports for departure and arrival, transporting our business person from the office or apartment to this airport, booking the flight to the chosen destination and transporting the business person from this airport to the business location.

A sample graph-based service decomposition is shown in figure 2. This decomposition will need a flight booking service as a single service for one of the components (i.e. travelling from airport to airport) and involve other services in the other steps. For instance the transportation from our business person's office to the airport can involve a taxi-service or public transportation which itself can be decomposed into a bus or a subway service. The transportation from the destination airport to the business location may again be of the simple kind like getting a cab, but can also involve complex tasks like renting a car. Complex tasks will again generally consist of several services that can be again disintegrated. Even in this simple example we can already identify twelve different ways of doing a business trip. Each way states a usage pattern that is categorized by the particular location and the time constraints, e.g. if you decide to take a flight and use the subway to the airport a subway service providing a line to the airport has to be chosen and a flight has to be booked that departs some time after the subway has arrived allowing you to check in. Such decompositions have been thoroughly investigated in planning algorithms in AI and knowledge representation [34]. Standard approaches to similar problems have been found to be very complex task. Thus deriving knowledge from service provided usage patterns, instead of relying on huge knowledge bases to segment every possible task seems sensible [5].

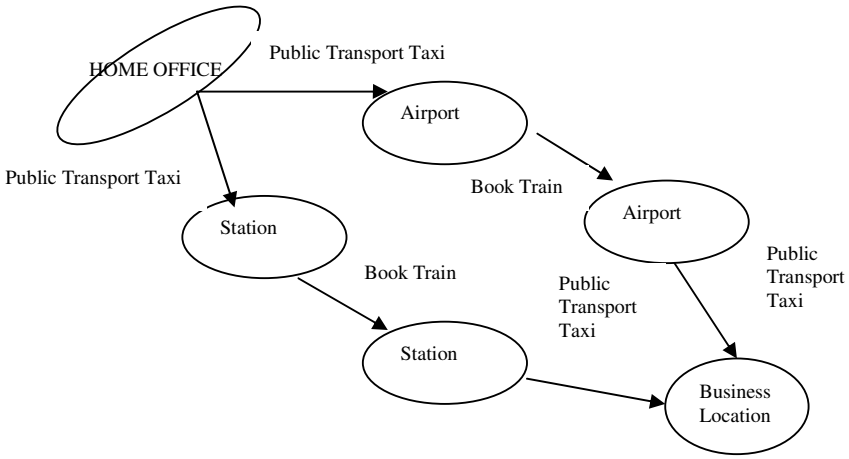

Figure 2. Decomposition of Services for Business Trips

As there is no standard solution to this problem, companies like e.g. travel agencies have certain general patterns or heuristics to deal with it. Starting with a general approach based on simple attributes like distance or duration of a journey they choose appropriate means of transportation or decide if booking a hotel is necessary, then working out the details. Web services are in this respect very alike. When designing a certain Web service, designers have quite specific ideas what potential the service should provide. Designers look forward to different possibilities for usage of the service and in well-defined services useful decompositions for certain typical queries or service requests will generally exist. These typical interactions for different user

groups are often referred to as usage knowledge. Usage knowledge depends on the intention of users. Generally speaking different intentions will need different patterns which is retrieved form usage knowledge .

Since different user groups will have at least some typical strategies to achieve the same kind of target, these groups can be clustered and assigned to different usage patterns. Consider for example a service for travelling. Of course the locations for departure and destination always have to be delivered and will differ from usage to usage, but possible decompositions would be taking the train or booking a flight. Generally speaking the train will be slower, but economical than the flight. Distinguishing only two groups of travellers we can assign some simple strategies. There are business travellers for whom time is probably a more important good than money. This can trigger default business travel patterns as for example the choice of avoiding the train or choosing business class tickets, if a minimization of flight time or changing of planes during the trip can be achieved. Also different general rules for specific domain knowledge apply. For instance the relaxation of constraints like arrival date and time is far trickier than a service would have to be with a usage pattern for tourists. Typical tourist patterns would probably be more worried about money and hence try to minimize the costs, e.g. taking the train would be an adequate choice, whereas flying business class could be considered less desirable. Besides, using a tourist pattern a service might be more bendable in relaxing time constraints.

When composing different services, usage patterns can be seen as a representation of the different potential or strategies that users apply to decompose their task. What sub goals are chosen has obviously a direct impact of the subsequent services that are performed to accomplish the complex task. Consider again our sample usage scenario from above. The complex transportation problem from the destination airport in Boston to the business location can be broken down into several sub goals. One option would be renting a car at the airport; another one would be taking a taxi or public transportation. Whatever way is subsequently chosen strongly depends on the specific user and thus is a matter for adequate personalization techniques using personal profiles stating preferred decomposition strategies or dislikes. These personalization strategies typically override more general terms of the pattern. For example, if a user suffers from fear of flying his or her personal profile would certainly state the dislike of flying though he or she might be part of the business travel pattern that prefers fast ways of travelling. In this case the specific personal profile would magnify the more general profile.

6. SUMMARY AND OUTLOOK

In this paper we have raised the question of Web service selection using Personalization techniques. Today Web service discovery and execution is performed using standards like UDDI, WSDL or ebXML, where limited semantic meaning is used in services description. Semantic Approaches like DAML-S or WSMF are first efforts that try to provide semantically rich service descriptions. But even letting aside the problems of incompatibility between different ontologies or missing concepts of interoperability, the human interaction with services poses severe problems due to the lack of personalization. Today's service variety offers users to choose between different services to perform their task. With the seam-less integration of networks this variety can be expected even to grow by orders of magnitude. The task of selecting an adequate service can quickly grow tedious, if all services that are listed under a certain description have to be compared manually for the final selection. And what is more, the final selection does not only depend on service parameters like execution costs or accuracy, but also depends on the usefulness of information that a service offers.

Interacting with a Web service a user has lots of expectations. Besides a (complex) goal there will be strategies to compose different services, certain preferred ways to achieve (sub-) goals, some hard constraints that have to be met and also a lot of user preferences or knowledge that is

implicitly given by previous service uses or domain knowledge. Finding useful means of personalization for each step we argued to divide a user's profile into three different parts:

i. The first part to decompose goals based on typical usage pattern anticipated by the service provider and helps to break down a complex task into preferred sub-goals that can be solved by simple Web services.

ii. The second part consists of user preferred service parameters and helps to choose between discovered services that advertise to perform the same task, but may differ in typical parameters like e.g. execution costs.

iii. The third part deals with the information that a service claims to provide. This part helps to discard services that can in principle handle the task, but don't provide any desirable objects.

Considering a sample interface we have shown, how to deal with the different user profiles and have presented an algorithm for the subsequent selection of appropriate services. This algorithm features an expansion of the service request by user-defined demands and wishes. Services not matching a certain profile are discarded on the fly and equally useful results of alternative services can be compared with respect to user preferences and strategies. Using cooperative answering techniques we also have showed how to get a better service selection including users' long-term profiles without risking running into empty result sets.

Aiming at improved usability of Web services our future work will focus on the subsequent refinement and integration of the presented ways of personalization into standardized interaction techniques. Investigations of repositories for user profiles, domain knowledge or usage patterns and the generation of new patterns have to be carried. We will also consider specific characteristics of our proposed personalization techniques and heuristics when dealing with different client devices. In the case of for instance mobile devices with limited capabilities the extension of our basic personalization techniques could pave the way to improved usability enabling a user-specific and context-aware selection of appropriate services.

REFERENCES

[1]. Ankolenkar, M. Burstein, S. Narayanan, et al. DAML-S: Web Service Description for the Semantic Web. . In Proc of Int. Semantic Web Conf. (ISWC'02), Sardinia, Italy, LNCS 2342, Springer, 2002.

[2]. Banerji, C. Bartolini, D. Beringer, et al. Web Services Conversation Language (WSCL) 1.0. http://www.w3.org/TR/wscl10/ (2002, accessed February 2011).

[3]. Dr. Mohammed T. Al-Sudairy, T. G. K Vasista, Semantic data integration approaches for e-governance, International Journal of Web & Semantic Technology (IJWesT) Vol.2, No.1, Jan 2011.

[4]. N.Sasikaladevi, Dr.L.Arockiam,Reliability Evaluation Model for Composite Web Services, International Journal of Web & Semantic Technology (IJWesT) Vol.1, No.2, April 2010.

[5]. C. W. Chan; Yao Peng; Lin-Li Chen, Knowledge acquisition and ontology modelling for construction of a control and monitoring expert system, International Journal of Systems Science, Volume 33, Issue 6 2002: pages 485 – 503.

[6]. Wen Jin. The Semantics and Approximation for the Skyline Operator, Phd Thesis Southeast University, Related articles ir.lib.sfu.ca/bitstream/1892/9216/1/etd2909.pdf (2007 accessed February 2011).

[7]. L Baresi. Tutorial – Towards Dynamic Web Services, Dynamic and Adaptive Composition of e-Services. Information Systems, Volume 6, Issue 3 2005: pages 143 – 163, www.irisa.fr/lande/lande/icse-proceedings/icse/p1067.pdf (2005 accessed February 2011).

[8]. E. Christensen, F. Curbera, G. Meredith, and S. Weerawarana. Web Services Description Language (WSDL) 1.1. http://www.w3.org/TR/2001/NOTE-wsdl-20010315 (2001 accessed February 2011).

[9]. W. Chu, H. Yang, K. Chiang, M. Minock, G. Chow, and C. Larson. CoBase: A Scalable and Extensible Cooperative Information System. Journal of Intelligent Information Systems (JIIS), 6(3), 2005: pp.223-259.

[10]. D. Connolly et al. DAML+OIL Reference Description. W3C Note, December 2001.

[11]. U Schmid. Inductive Rule Learning on the Knowledge Level, Elsevier-Artificial Intelligence, Volume 18, 2010, pp.87-127.

[12]. Doan, J. Madhavan, P. Domingos : Learning to map between ontologies on the semantic web. In Proc. of the Int. World Wide Web Conf. (WWW 2002),, Honolulu, Hawaii, USA, 2002: pp. 662-673.

[13]. E-Speak Architectural Specification Release A.0. http://www.e-speak.hp.com/media/a0/architecturea0.pdf (2004 accessed February 2011).

[14]. ebXML. ebXML Web Site. http://www.ebXML.org (2005 accessed January 2011).

[15]. Expedia Travel Portal. http://www.expedia.com (2007 accessed February 2011).

[16]. D. Fensel and C. Bussler. The Web Service Modeling Framework WSMF. Appeared in Electronic Commerce Research and Applications. http://www.cs.vu.nl/~dieter/ wese/wsmf.paper.pdf (2007 accessed February 2011).

[17]. R Pandey. Ontology Description using OWL to Support Semantic Web. International Journal for Computer Application (IJCA), Vol.14, No.6, Mar 2011.

[18]. W. Kießling, G. Köstler: Preference SQL - Design, Implementation & Experiences. In Proc. of the Int. Conf. On Very Large Databases(VLDB). Hong Kong, China, 2002.

[19]. Bhaskar Kapoor and Savita Sharma. A Comparative Study Ontology Building Tools for Semantic Web Applications. International journal of Web & Semantic Technology (IJWesT) Vol.1, Num.3, July 2010.

[20]. F. Leymann. Web Services Flow Language (WSFL 1.0). http://www-4.ibm.com/software/solutions/webservices/pdf/

[21]. WSFL.pdf (2001 accessed January 2011).

[22]. R. S. Michalski, J. G. Carbonell, and T. M. Mitchell. *Machine Learning: An Artifical Intelligence Approach*. Morgan Kaufmann, 1986.

[23]. CF Dorneles, Approximate data instance matching: a survey, In Proceeding of ACM Trans. Off Inf Syst 6(3):187–214, Jan 2010.

[24]. NET Passport. http://www.passport.com , (2002, accessed January 2011).

[25]. T. M. Mitchell. Generalization as Search. Artifical Intelligence, 18(2), 1982: pp.203-226.

[26]. S. McIlraith, T. Son, H. Zeng: Semantic Web Services. In IEEE Journal on Intelligent Systems, IEEE, 2001: pp. 46-53.

[27]. J. Minker. An Overview of Cooperative Answering in Databases. In Proceedings of the 3rd International Conference on Flexible Query Answering Systems (FQAS), 1998: pp. 282-285.

[28]. S. Narayanan and S. McIlraith. Simulation, Verification and Automated Composition of Web Services. In Proc of Int. World Wide Web Conf. (WWW 2002), Honolulu, Hawaii, USA, 2002: pp. 77-88.

[29]. F. Patel-Schneider and D. Fensel. Layering the Semantic Web: Problems and Directions. In Proc of Int. Semantic Web Conf. (ISWC$(B02(B), Sardinia, Italy, LNCS 2342, Springer, 2002.

[30]. P. Pires, M. Benevides and M. Mattoso. Building Reliable Web Services Compositions. In Proc. of the Int. Workshop on Web Services: Research, Standardization and Deployment (WS-RSD), Erfurt, Germany, 2002: pp. 551-562.

[31]. SOAP Protocol. http://www.w3.org/2000/xp/Group (2000 , accessed December 2010).

[32]. Liberty Alliance. http://www.projectliberty.org (2002, accessed January 2011).

[33]. S. Thatte. XLANG: Web Services for Business Process Design. http://www.gotdotnet.com/team/xml_wsspecs/xlang-c/default.htm (2001, accessed December 2010).

[34]. UDDI. The UDDI Technical White Paper. http://www.uddi.org (2002, accessed January 2011).

[35]. M. Vilain. Getting Serious about Parsing Plans: a Grammatical Analysis of Plan Recognition. In Proc. of the Nat. Conf. on Artificial Intelligence (AAAI-90), Boston, USA, 1990: pp. 190-197.

[36]. M. Wagner, W.-T. Balke, R. Hirschfeld and W. Kellerer. A Roadmap to Advanced Personalization of Mobile Services. In Proc. of the Int. Federated Conferences DOA/ODBASE/ CoopIS (Industry Program). Irvine, CA, 2002.

Authors

Shailesh P. Khapre is Mtech (CSE) student in Pondicherry University. His area of interest includes Web Service Selection, Service Personalization, Cloud Computing and Software Testing. Currently he is working on user preference based web service Selection & Composition.

D. Chandramohan is Mtech (CSE) student in Pondicherry University. His area of interest includes Distributed Web Service, Web Service (Evaluation) Testbed, Software Metrics and Cloud Computing. Currently he is working on Designing of Testbed for Distributed web service environment.