Designing a Collaborative Middleware for Semantic and User-aware Service Composition

Giacomo Cabri, Riccardo Martoglia, Franco Zambonelli
Università di Modena e Reggio Emilia, Italy,
Email: name.surname@unimore.it

***Abstract*—The large number of available services, provided by different means such as the Web, smartphone apps, and wearable devices, provides users a valuable support for their everyday activities, but at the same time introduces the need for a tailored choice and exploitation of them. Several approaches have been proposed that take into account users' preferences, but a comprehensive user-aware approach is still missing.**

In a previous work we proposed an approach that addressed the user-aware composition of services; in this paper we propose to extend the previous approach by considering also semantic techniques and simple collaboration aspects. So, we propose the definition of a middleware for composing and exploiting services that exhibits some key features: (i) it considers the profile of the users that exploit the service to choose appropriate services for them, (ii) it exploits techniques of semantic similarity between user and service descriptions to make the choice more effective, and (iii) it enables the collaboration among users. By means of a case study we present a possible scenario that can take advantage of our middleware, and we show how it can be exploited.

Keywords*—*services, user-awareness, context

I. Introduction

Our everyday life is more and more supported by different kinds of services, which are provided by internet connected electronic devices. Such services are typically exploited by *apps* or via a web browser, both on smartphones; but new means are appearing, such as smart objects and wearables devices. Many "spaces" are becoming "smart", like offices, streets, hospital, airports, homes and shops equipped with sensing systems to help people find relevant information quickly and use services comfortably.

The number and kind of services are going to increase in the future, and the related risk is that the user is overwhelmed by offers of services, which are often not interesting. To address this situation, many researchers have proposed to develop applications with user-awareness capabilities [1], [2], [3]. A user-aware application recognizes the context in which the user is performing an activity by means of that application and exploits contextual information to adapt its behaviour.

Moreover, often a single service satisfies only a part of the user's request, leading to a need for *composing* different services. In [4] we have addressed this issue, proposing an approach to achieve a user-aware composition of different services, evolving the idea of single user-aware services.

In this paper we aim to go further, *extending* and *detailing* the framework sketched in the previous paper, considering *semantic similarities* by adding appropriate techniques, and addressing the *collaboration* issues in the presented scenario. The fact that services can provide a form of collaboration among users, makes the situation much more dynamic, and requires a much more sophisticated matching mechanism between keywords, also taking into consideration their semantics.

For instance, consider an e-commerce service that provides discounts to group purchases; in this case, the larger the number of users that decide to exploit a service, the cheaper will be the cost of the service itself. On the one hand, the supermarket can provide different services (for instance, selling, payment, delivery, . . .) to the single user; on the other, the group purchase emerges from the collaboration among the customers: considering their profile, the system could suggest goods to buy or delivery means that both meet the users' expectations and enable a discount thanks to the group purchase.

The user-aware collaboration among different users introduces also another issue: users can have similar preferences expressed by different keywords; in this case, the use of semantic mechanisms is fundamental to accomplish our task.

Starting from these considerations, in this paper we propose an approach for collaborative services that enables user-awareness and exploits semantic mechanisms to enact it. This approach is developed in the context of the AMBIT project[1], whose aim is to study and develop a software architecture for building context-dependent applications and systems.

The remainder of paper is organized as follows. First, we introduce some related work (Section II). Then, we present the SAPERE (Self-Aware Pervasive Service Ecosystems) infrastructure, which is exploited in our work (Section III). The considered case study is reported in Section IV. In Section V we introduce our approach to compose services taking into consideration the user profile. Finally, we conclude the paper and sketch some future work (Section VI).

II. Related Work

User-awareness or more in general context-awareness and context-dependent service delivery have been addressed in an already impressive body of work. These efforts are largely motivated by the astounding growth of the mobile device market and the need to support the evolution of the traditional Web into the so-called "Web of Things and Services". For some comprehensive surveys the reader is referred to [5], [6], [7] and the references contained therein.

In the context of this huge active area, our research interests are primarily directed towards providing better customized

[1] http://didattica.agentgroup.unimore.it/ambit/

service compositions and services. There are a handful of research contributions that are relevant to this narrow area.

The approach [8] takes different types of context information into account (physical, computational, user context) and tackles the problem of modeling and representation from within the perspective of automated reasoning. The rationale is that modeling real-life situations requires the ability to process basic context facts and reasoning makes it possible to attain pieces of information that are appropriate for use by context-aware applications. This highly cited survey is also important since it covers some prominent approaches to context modeling (object-role based, spatial models, ontology-based).

In [9], the authors propose a general architectural framework for context data management, which include separate components for context source and context provider modeling. To move closer towards interoperability, they also propose the definition of standard languages for data context representation (ContextML) and access (Context Query Language). The proposed architecture is interesting for our purpose, but our aim is to include semantic information about the context.

The paper [10] surveys various research works related to context modeling and awareness within the Context-ADDICT project of the Politecnico di Milano (see http://poseidon.ws.dei.polimi.it/ca/). They propose to design a context management system to be placed aside what they call the "operational system". While the latter is application dependent, the context management system is not, and it exhibits a hierarchical structure in terms of observable (i.e., external) parameters that have a symbolic internal representation within a context schema. We will consider this separation of concerns for building the bases of our user-aware service composer.

In [11], the authors report the result of a study on various context modeling and management approaches. They outline four main research challenges that have to be tackled to support context-awareness in the light of the Smart Internet. They propose an operational definition of a context as well as a *context taxonomy* based on previous classifications of context information, which includes the following five fundamental context categories: individuality, time, location, activity and relation. The idea of a taxonomy turns out to be very useful to our semantic approach. In [12], the authors propose a context-based approach for service discovery. The focus on services makes this work interesting for our purpose, even if we do not focus on the discovery. We will evaluate their formal definition of the context and consider it in our work.

As to Web collaborative services, most of the research work focus on enabling collaboration among users [13], on "collaborative filtering" [14], [15], and on "collaborative intrusion detection" [16]. Differently from them, our approach aims at injecting user-awareness in collaborative services, enabling a more suited collaboration and providing matching between services and users, and between users with similar profiles.

III. The SAPERE Approach to Service Composition

A. The SAPERE Model

SAPERE founds on the consideration that the large multitude of ubiquitous services that will soon enrich our lives will make it suitable to model the set of such services as a sort of distributed pervasive service *ecosystem* [17]. It conceptually models such pervasive ecosystem as a virtual *spatial environment*[18], laid above the actual network of devices infrastructure. The environment acts as a sort of shared space in which all service components situate, and the environment itself takes care of mediating all interactions. In other words, the spatial environment represents the ground on which services of different species indirectly interact and combine with each other. Such interactions take place in respect of a limited set of basic interaction laws (also called "eco-laws", due to their nature-inspired origins), and typically accounting on the spatial and contextual relationships between services.

SAPERE adopts a common modeling and treatment for the *service components* populating the ecosystem. Each of them has an associated semantic representation which we call "LSA" (*Live Semantic Annotation*, a list of properties and characteristics for each service), to be injected in the spatial environment as it were a sort of shared spatial memory.

The LSA is a basic ingredient for enabling dynamic environment-mediated interactions between services. In fact, to account for the high dynamics of the scenario, SAPERE defines LSAs as living, active entities. LSA are tightly associated to the service component they describe, and capable of reflecting its current situation and context. This is opposed to, say, the more static nature of WSDL descriptions. This supports semantic and context-aware interactions both for service aggregation/composition and for data/knowledge management. The *eco-laws* define the basic interaction policies among the LSAs of the various services of the ecosystem. The idea is to enforce on a spatial basis, and possibly relying on diffusive mechanisms, dynamic composition of data and services by composing their LSAs and exchanging data via them. Data and services, as represented by their associated LSAs, will be like chemical reagents, and interactions and compositions will occur via chemical reactions, relying on semantic matching between LSAs.

In this paper, we will not go into details about the specifics of all the SAPERE eco-laws; anyway, we want to emphasize that the advanced forms of adaptive matching between LSAs that they enforce can make it possible to dynamically compute, at any time and for every service of the ecosystem, the list of services potentially matching with other services. Adaptivity in SAPERE is not in the capability of individual services, but in the overall self-organizing dynamics of the service ecosystem as a whole. In particular, adaptivity will be ensured by the fact that any change in the system, as well as any change in its services or in the context of such services, will reflect in the firing of new eco-laws, thus possibly leading to the establishment of new compositions or aggregations, and/or in the breaking of some existing service compositions.

B. The SAPERE Infrastructure

An infrastructure [19] which reifies the SAPERE environment in terms of a lightweight software supports the execution of SAPERE applications and enables a SAPERE node to be installed in tablets and smartphones. Each SAPERE node wishing to participate to the SAPERE ecosystem should host a local tuple space [20], to act as a local repository of LSAs for local services, and a local eco-laws engine. The LSA-space of

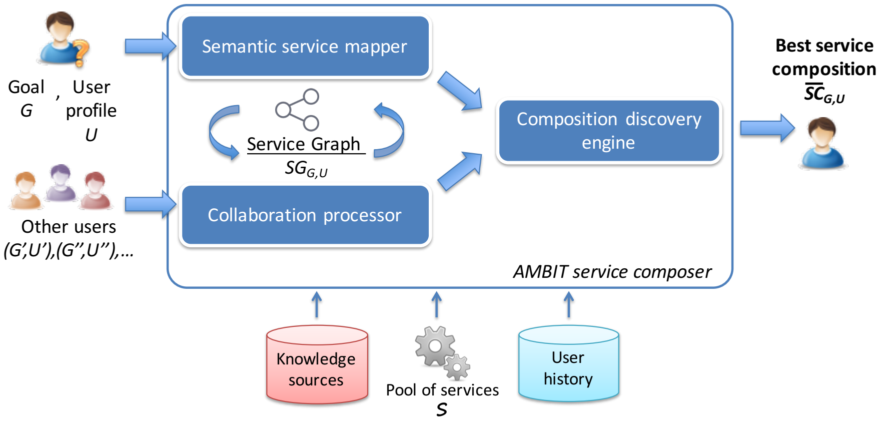

Fig. 1. Overview of the architecture of the AMBIT service composer

each node is connected with a limited set of neighbor nodes based on spatial proximity relations. From the viewpoint of individual services, the infrastructure provides an API to access the local LSA space, to advertise themselves (via the injection of an LSA), and to support the services' need of continuously updating their LSAs. In addition, such API enables services to detect local events (as the modifications of some LSAs) or the enactment of some eco-laws on available LSAs.

SAPERE nodes realize eco-laws by means of a set of rules embedded in them. For each node, the same set of eco-laws applies to rule the dynamics between local LSAs (in the form of bonding, aggregation, and decay) and those between non-local LSAs. In the latter case, this is done via the spreading eco-law that can propagate LSAs from a node to another, in order to support distributed service interactions and composition.

IV. Case Study

A typical example involving service composition is an e-commerce transaction. Typically, this involves a service for the selection of the good to acquire, providing information about the product and its price, followed by services to perform the payment transaction, and eventually a service for the delivery of goods. For each of these services, multiple versions can be provided (e.g., credit card vs paypal, or delivery via different courier companies).

Currently, most online e-commerce web site leave little choice on the specific services that will eventually provide the complete composite one, or at most leave users in charge of explicitly selecting them over the course of the transaction. However, there is an increasing diversity of services providing equivalent functions to choose from, which make leaving the choice up to the user increasingly difficult. Also, in service provisioning, there is also the need to accommodate specific user preferences and to account from the social and physical context in which the services are requested. Accordingly, even simple services compositions such as the above described e-commerce transaction will have to involve a dynamic automatic selection of the services, accounting for user preference and for the current user context.

For instance consider an e-commerce site that is promoting group discounts, e.g., discounting prices of products whenever a minimum amount of users willing to buy the same product is reached. And, similarly, delivery services that provide discounts for groups of buyers willing to deliver in the same city. In this specific case, a user may wish that the services exploited to select products, and later to deliver the bought products, take advantage of the possibility of group discount. However, to this end, it is necessary to have some support that can automatically select where to buy and how to deliver depending on the amount of other persons concurrently (i.e., within a specific time interval) wishing to buy the same products, and on the amount of persons in the same city currently wishing to deliver goods. In other words, there is a need for dynamic service composition mechanisms capable of accounting for individual user preference and for their current social and physical context.

V. Towards Service Composition based on User Profile

In this section we present our middleware, the AMBIT service composer, which relies on the SAPERE infrastructure, and our approach to user-aware service composition.

A. *Overview of the AMBIT Service Composer*

The AMBIT service composer, whose architecture is shown in Fig. 1, consists of three main modules: the *Semantic service mapper*, the *Collaboration processor* and the *Composition discovery engine*.

The *Semantic service mapper* exploits the SAPERE infrastructure, embedding eco-laws and capable of digesting the LSAs of the different services of an ecosystem. In reaction to a request of a user characterized by a user profile U (i.e., its preferences and context), the request is translated into a goal G, which from within SAPERE takes the form of an LSA representing specific desirable features of a service (e.g. an object on sale, payment options, and so on). By interacting with the *Collaboration processor*, which takes into consideration the profiles and goals $\{(G', U'), (G'', U''), \dots\}$ of other concurrent users $\mathcal{U}$, a dynamic graph of service connections (***service graph***) $SG_{G,U}$ is constructed and continuously kept updated; a change in the collaboration can modify the weights of the graph. Then, the *Composition discovery*

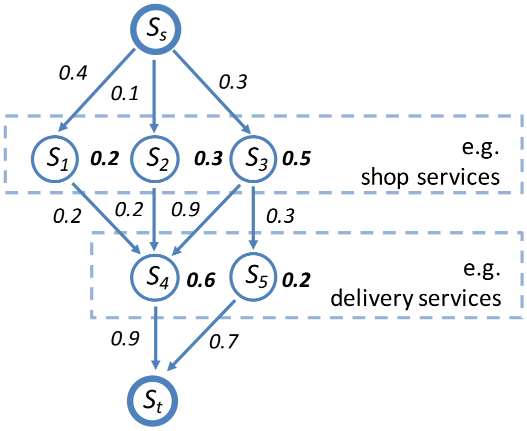

Fig. 2. An example of a service graph

engine analyzes the graph and has the goal of finding the best service composition $\overline{SC}_{G,U}$ for user profile U and goal G.

Let us now analyze more in depth how the service graph is constructed and how it is exploited in order to achieve this.

B. Service Graph

First of all, we assume a goal G to be characterized by a set of keywords, i.e., $G = \{k_i^G\}_{i=1,\ldots,m}$. Similarly, the user profile is constituted by a set of keywords $U = \{k_i^U\}_{i=1,\ldots,n}$. The way keywords k_i^U are derived is beyond the scope of this paper, anyway they can be determined by using standard text analysis techniques, such as the ones described in [21], operating on the environment, user, and history data of the profile. For instance, a goal G may include specific keywords such as "gym suit" or "cardigan"; a profile U could include keywords on preferences (i.e. "sport"), on context (i.e. "Italy"), and so on[2]. Also, we consider a pool of available services $\mathcal{S}$; each service $S \in \mathcal{S}$ is defined as $S = \{k_i^S\}_{i=1,\ldots,l}$ a set of keywords k_i^S derived from the service description that characterizes the service itself. For instance, a European service selling clothes could include such keywords as "sportswear", "Europe", "pullovers", etc.

A service graph $SG_{G,U}$ is basically a weighted network of services and service connections built w.r.t. user goal G and profile U and services $\mathcal{S}$. More specifically, $SG_{G,U}$ is defined as a connected directed labeled graph $SG_{G,U} = (\overline{\mathcal{S}}, C, w_{G,U}, w_U)$ where $\overline{\mathcal{S}} \subseteq \mathcal{S}$ is a set of nodes (services), $C \subseteq S \times S$ is a set of directed arcs (service connections) and $w_{G,U} : \mathcal{S} \rightarrow [0,1]$ and $w_U : C \rightarrow [0,1]$ are two functions mapping nodes and arcs to their weights, respectively. $SG_{G,U}$ includes a source S_s and sink S_t corresponding to fictitious service nodes.

The weights (described in detail in Sections V-C and V-D) reflect the suitability of specific services and service connections in SG_{G_U} to goal G and user profile U, also taking into account collaboration with concurrent users and past user history. In Fig. 2 we report an example of service graph related to the case study previously introduced.

C. Service weights

The idea behind the service weights $w_{G,U}(S)$ in $SG_{G,U}$ is to quantify the relevance and suitability of each service S in $SG_{G,U}$ w.r.t G, S and the amount of concurrent users which are currently exploiting the service[3]. In the example of Fig. 2, the weights of shop service $S1$, $S2$ and $S3$ depend on the result of the match between the user's interests and the goods provided by the single services, and by the number of users interested in those services; the delivery service $S4$ scores 0.6 because it provides a physical shop near the city of the current user, while $S5$ does not. Generally speaking, the weights will be computed taking into account:

- the similarity between goal G and service S;
- the similarity between user profile U and service S;
- the collaboration ratio of the service S.

The service mapper and collaboration processor do this by means of the following formula:

$$w_{G,U}(S) = \alpha \cdot dsim(G,S) + \beta \cdot dsim(U,S) + \gamma \cdot cr(S) \quad (1)$$

where $dsim \in [0,1]$ is a similarity function computed between two descriptions (i.e. keyword sets) and cr is a function expressing the collaboration ratio of a given service (see later). α, β and γ ($\alpha+\beta+\gamma = 1$) are tunable positive parameters that can be freely adjusted in order to change the relative influence of G, U and collaboration.

Semantic similarity. In order to allow an effective matching between the descriptions of goals G, user profiles U and services S, we adopt a *semantic*, rather than a "syntactic" approach where keywords are matched on the basis of their meaning. In particular, we exploit information coming from thesauri such as WordNet [22], which will be considered in the following, and Expert System's Cogito[4]. Building on previous research on text retrieval in a software engineering and/or user-aware context [23], [21], [24], we define the semantic similarity $dsim$ between two descriptions D_1 and D_2 as:

$$dsim(D_1, D_2) = \frac{\sum_{k^{D_1} \in D_1} \max_{k^{D_2} \in D_2} ksim(k^{D_1}, k^{D_2})}{|D_1|} \quad (2)$$

$ksim$ is computed by means of Equation (3) which considers *synonyms* (thus, implicitly equal keywords) and *semantically related keywords*:

$$ksim(k_i, k_j) = \begin{cases} 1, & \text{if } k_i = k_j \text{ or } k_i \ SYN \ k_j \\ r, & \text{if } k_i \ REL \ k_j \\ 0, & \text{otherwise.} \end{cases} \quad (3)$$

Note that the case of maximum similarity (i.e. value 1) holds when the two keywords are synonyms (SYN relation). Moreover, when the two keywords are in some way related (REL relation) from a semantic point of view (i.e., broader/narrower keywords etc.), the formula provides a similarity value of r, where $0 < r < 1$ is a constant which can be adjusted on a global basis by the user in order to increase or reduce the influence of related words in the similarity

[2] In order to facilitate semantic keyword matching, keywords are also associated to specific categories such as "preference", "context", "region",etc; for simplicity sake, in this paper we will deal with plain keyword sets.

[3] In this paper we report a very simple case of collaboration, but of course more complex ones can be considered.

[4] http://www.expertsystem.com/it/cogito/

framework. For example, going back to our example (Fig. 2), by means of these semantic relations the framework can easily match a user goal for a “cardigan” to a service $S3$ selling “pullovers” (synonym) and, to a lesser extent (lower similarity), to a service $S2$ selling “sweaters” (related term), even if no common keywords are present in their descriptions.

Exploiting the relations between keywords coming from the WordNet thesaurus offers an effective way to compute semantic relatedness. Indeed, in WordNet terms are associated to one or more different meanings (or senses), and each term is then connected to other terms meanings by hypernym (i.e., “is-a”) relations[5] [22]. We adopt one of the most widely used methods in knowledge management, relying on the hypernym relations:

$$k_i \; REL \; k_j \Longleftrightarrow hsim(k_i, k_j) \geq Th \tag{4}$$

$$hsim(k_i, k_j) = \begin{cases} -ln\frac{len(k_i,k_j)}{2H}, & \text{if } \exists \; lca(k_i, k_j) \\ 0, & \text{otherwise.} \end{cases} \tag{5}$$

In our case, two keywords are semantically related if their hypernym similarity $hsim$ exceeds a given threshold Th (Equation 4). In particular, the $hsim$ shown in (5) derives from [25] and computes a score which is inversely proportional to the length of the shortest path connecting the (senses of the) two keywords. H is a constant representing the maximum depth of the hypernym tree, which for WordNet is defined as 16. On the other hand, the similarity is 0 if the two keywords are not connected in the WordNet hypernym structure.

Collaboration ratio. The idea behind the quantification of a *collaboration ratio cr* for a given service S in $SG_{G,U}$ is to express the fact that the more concurrent users are exploiting S the more valuable it will be for user goal G and profile U. For instance, in a collaborative shop scenario, the more users are currently buying a given good, the more favorable will be its acquisition. We express this by means of the following ratio:

$$cr(S) = \frac{|\{\overline{SC_{\mathcal{U}}} : S \in (S_x, S_y), (S_x, S_y) \in \overline{SC_{\mathcal{U}}}\}|}{max\_u_S} \tag{6}$$

Equation (6) represents the number of best service compositions (suggested by the service composer to other concurrent users $\mathcal{U}$) currently involving S, divided by the maximum number of users max_u_S supported / required by S. For instance, max_u_S could represent the maximum number of buyers for a given good. The *collaboration processor* module dynamically computes all the required collaboration ratios in the service graph, and keeps them updated as new users / requests reach the system.

D. Service connection weights

Service connection weights $w_U(S_x, S_y)$ complete the $SG_{G,U}$ service graph by taking into account the suitability of each service connection (S_x, S_y) (i.e. each arc of our service graph) w.r.t. user profile U and past *user history*. For instance, in the example of Fig. 2, the shop service $S3$ has an agreement with the delivery service $S4$ to reduce the delivery cost, so the user has frequently used the couple of services and the arc connecting them has a high score.

To achieve this, the service composer stores the history of service compositions assigned to users in past requests and estimates $w_U(S_x, S_y)$ as the *probability* $P_U(S_y|S_x)$ that a given service connection (S_x, S_y) is suitable for a given user profile U:

$$w_U(S_x, S_y) = P_U(S_y|S_x) \tag{7}$$

But how can $P_U(S_y|S_x)$ be quantified? Basically, given a service S_x, we find the most likely service S_y that could follow for user profile U by working at the level of the single keywords composing profiles. For a generic user $U = \{k_1, \ldots, k_n\}$ and any $I \subseteq \{1, \ldots, n\}$, let $count_I(S_x)$ and $count_I(S_x, S_y)$ denote the number of times users characterized (possibly among others) by the set of keywords $\{k_i\}_{i \in I}$ have been successfully serviced by service S_x and service connection (S_x, S_y), respectively. The probability $P_U(S_y|S_x)$ of successful service connection (S_x, S_y) for user U can then be estimated using the well-known principle of inclusion-exclusion[6]:

$$P_U(S_y|S_x) \approx \frac{\sum_{e=1}^{n} (-1)^{e-1} \sum_{I \subseteq \{1,\ldots,n\}, |I|=e} count_I(S_x, S_y)}{\sum_{e=1}^{n} (-1)^{e-1} \sum_{I \subseteq \{1,\ldots,n\}, |I|=e} count_I(S_x)} \tag{8}$$

Since computing Eq. 8 exactly requires exponential work, in our approach we will recur to approximation (see, e.g., [26]) or even heuristic algorithms.

E. Finding best service composition

Given the service graph $SG_{G,U}$ as provided by the semantic service mapper and collaboration processor (see previous sections), the goal of the *composition discovery engine* is to find, among all sequences $SC_{G,U}$ of consecutive connection arcs, the “best” service composition(s) $\overline{SC}_{G,U}$.

To this end, we define a ***score*** of a service composition $SC_{G,U}$ by composing the weights of its single service connections (and their services) as defined in Eqs. 1 and 7:

$$score(SC_{G,U}) = \prod_{(S_x,S_y) \in SC_{G,U}} w_U(S_x, S_y) \cdot w_{G,U}(S_y) \tag{9}$$

In particular, the product privileges the shortest sequences (eg. a composition of a very large number of suitable services is not expected to be equally suitable). In this way, finding the “best” service composition $\overline{SC}_{G,U}$ becomes a matter of finding the sequence of service connections maximizing the score in Eq. 9. Since the service graph is a DAG (Directed Acyclic Graph) and the “score” of service composition cannot but decrease when extending a path, this computation can be efficiently performed (in linear time) using a slightly modified version of Dijkstra algorithm.

[5] We recall that k_i is said to be a hypernym of k_j if there exists a k_i's meaning that includes (i.e., is a hypernym) of a meaning of k_j: for instance, “electronic device” is a hypernym of “computer”.

[6] We adopt the well known Markov chain approximation: in our context, the probability of choosing the next service depends only on the preceding service and not on the whole sequence of services that preceded it.

For instance, for our example in Fig. 2, $\overline{SC}_{G,U} = \{(S_s, S_3), (S_3, S_4), (S_4, S_t)\}$, $score(\overline{SC}_{G,U}) = (0.3 \cdot 0.5) \cdot (0.9 \cdot 0.6) \cdot (0.9 \cdot 1.0)$ (for the computation, the weights of fictitious services S_s and S_t are considered unitary).

VI. Conclusions

In this paper we have presented a user-aware middleware for service composition, which exhibits three key features: it considers the profile of users that exploit the service to choose appropriate services, it exploits semantic similarity techniques to improve the match between the users' profile and the services' descriptions; it supports the collaboration among users that exploit services.

The proposed middleware is based on (and extends) the SAPERE infrastructure to compose services. To choose the most appropriate set of services, our middleware builds a graph whose nodes are the services and whose arcs represent the composition between them. The definition of weights for both nodes and arcs allows us to evaluate the best path inside the graph, which represents the composition of services that better suits user's profile. The collaboration among users implies that such a graph (i.e. its weights) can change during runtime, so our approach takes this into consideration and dynamically evaluates the score of the service compositions.

In future work, we will investigate further possibilities for modeling user-awareness, including categorizing users w.r.t. ontologies in order to further personalize their results (e.g. by adapting the techniques described in [27]). Moreover, we aim at producing an effective implementation of the middleware, to be tested by means of some case studies in different application fields. For instance, a test can involve true users that define their preferences and are asked to compare the sets of services resulting from applying our approach w.r.t. not applying it.

Acknowledgment

This work was supported by the project "Algorithms and Models for Building context-dependent Information delivery Tools" (AMBIT) co-funded by Fondazione Cassa di Risparmio di Modena (SIME 2013.0660). Website: http://www.agentgroup.unimore.it/ambit/

References

[1] M. Baldauf, S. Dustdar, and F. Rosenberg, "A survey on context-aware systems," *International Journal of Ad Hoc and Ubiquitous Computing*, vol. 2, pp. 263–277, 2007.

[2] G. Cabri, L. Leonardi, M. Mamei, and F. Zambonelli, "Location-dependent Services for Mobile Users," *IEEE Transactions on Systems, Man, and Cybernetics-Part A: Systems And Humans*, vol. 33, no. 6, pp. 667–681, 11 2003.

[3] F. Zambonelli, N. Bicocchi, and D. Fontana, "Towards a human-aware operating system," in *BODYNETS 2014, London, Great Britain, September 29 - October 1, 2014*, 2014.

[4] G. Cabri, M. Leoncini, R. Martoglia, and F. Zambonelli, "Towards User-aware Service Composition," in *Proceedings of the 2nd EAI International Conference on Nature of Computation and Communication*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering (LNICST). Springer, 3 2016.

[5] A. G. de Prado and G. Ortiz, "Context-aware services: A survey on current proposals," in *SERVICE COMPUTATION 2011, The 3rd Int. Conf. on Advanced Service Computing*, 2011, pp. 104–109.

[6] G. M. Kapitsaki, G. N. Prezerakos, N. D. Tselikas, and I. S. Venieris, "Context-aware service engineering: A survey," *J. Syst. Softw.*, vol. 82, no. 8, pp. 1285–1297, Aug. 2009.

[7] Q. Z. Sheng and S. Dustdar, "Introduction to special issue on context-aware web services for the future internet," *ACM Trans. Internet Technol.*, vol. 11, no. 3, pp. 9:1–9:2, Feb. 2012. [Online]. Available: http://doi.acm.org/10.1145/2078316.2078317

[8] C. Bettini, O. Brdiczka, K. Henricksen, J. Indulska, D. Nicklas, A. Ranganathan, and D. Riboni, "A survey of context modelling and reasoning techniques," *Pervasive and Mobile Computing*, vol. 6, no. 2, pp. 161–180, 2010, context Modelling, Reasoning and Management.

[9] P. Falcarin, M. Valla, J. Yu, C. A. Licciardi, C. Frà, and L. Lamorte, "Context data management: An architectural framework for context-aware services," *Serv. Oriented Comput. Appl.*, vol. 7, no. 2, pp. 151–168, Jun. 2013. [Online]. Available: http://dx.doi.org/10.1007/s11761-012-0115-1

[10] C. Bolchini, G. Orsi, E. Quintarelli, F. A. Schreiber, and L. Tanca, "Context modeling and context awareness: steps forward in the context-addict project," *Bulletin of the Technical Committee on Data Engineering*, vol. 34, pp. 47–54, 2011.

[11] N. M. Villegas and H. A. Müller, in *The Smart Internet*, ser. Lecture Notes in Computer Science, 2010, vol. 6400, pp. 289–318.

[12] P. Venkataram and M. Bharath, "A method of context-based services discovery in ubiquitous environment," in *Context-Aware Systems and Applications*. Springer, 2014, pp. 260–270.

[13] D. K. Chiu, S. Cheung, H.-f. Leung, P. C. Hung, E. Kafeza, H. Hu, M. Wang, H. Hu, and Y. Zhuang, "Engineering e-collaboration services with a multi-agent system approach," *International Journal of Systems and Service-Oriented Engineering*, vol. 1, no. 1, pp. 1–25, 2010.

[14] X. Su and T. M. Khoshgoftaar, "A survey of collaborative filtering techniques," *Advances in artificial intelligence*, vol. 2009, p. 4, 2009.

[15] M. D. Ekstrand, J. T. Riedl, and J. A. Konstan, "Collaborative filtering recommender systems," *Foundations and Trends in Human-Computer Interaction*, vol. 4, no. 2, pp. 81–173, 2011.

[16] H. T. Elshoush and I. M. Osman, "Alert correlation in collaborative intelligent intrusion detection systemsa survey," *Applied Soft Computing*, vol. 11, no. 7, pp. 4349–4365, 2011.

[17] F. Zambonelli, A. Omicini, B. Anzengruber *et al.*, "Developing pervasive multi-agent systems with nature-inspired coordination," *Pervasive and Mobile Computing*, vol. 17, pp. 236–252, 2015.

[18] D. Weyns, A. Helleboogh, T. Holvoet, and M. Schumacher, "The agent environment in multi-agent systems: A middleware perspective," *Multiagent and Grid Systems*, vol. 5, no. 1, pp. 93–108, 2009.

[19] G. Castelli, M. Mamei, A. Rosi, and F. Zambonelli, "Engineering pervasive service ecosystems: the sapere approach," *ACM Transactions on Autonomous and Adaptive Systems (TAAS)*, vol. 10, no. 1, p. 1, 2015.

[20] D. Gelernter, "Generative communication in linda," *ACM Transactions on Programming Languages and Systems*, vol. 7, no. 1, pp. 80–112, Jan. 1985.

[21] R. Martoglia, "AMBIT: semantic engine foundations for knowledge management in context-dependent applications," in *SEKE*, 2015, pp. 146–151.

[22] G. A. Miller, "Wordnet: A lexical database for english," in *HLT*, 1994.

[23] S. Bergamaschi, R. Martoglia, and S. Sorrentino, "Exploiting semantics for filtering and searching knowledge in a software development context," *Knowledge and Information Systems*, vol. 45, no. 2, pp. 295–318, 2015.

[24] D. Beneventano, S. Bergamaschi, and R. Martoglia, "Exploiting semantics for searching agricultural bibliographic data," *Journal of Information Science*, 2015.

[25] C. Leacock and M. Chodorow, "Combining local context and wordnet similarity for word sense identification," in *WordNet: An electronic lexical database*, 1998.

[26] J. Kahn, N. Linial, and A. Samorodnitsky, "Inclusion-exclusion: Exact and approximate," *Combinatorica*, vol. 16, no. 4, pp. 465–477, 1996.

[27] F. Grandi, F. Mandreoli, and R. Martoglia, "Issues in personalized access to multi-version xml documents," in *Open and Novel Issues in XML Database Applications: Future Directions and Advanced Technologies, Eric Pardede (Ed.), IGI Global*, 2008, pp. 199–230.