Exploratory Search of Web Data Services Based on Collective Intelligence

Devis Bianchini(✉), Valeria De Antonellis, and Michele Melchiori

Department of Information Engineering, University of Brescia,
Via Branze, 38, 25123 Brescia, Italy
{devis.bianchini,valeria.deantonellis,michele.melchiori}@unibs.it

Abstract. Developers of data-intensive web applications benefit from the integration of data sourced from the web. Web data services are solutions off-the-shelf, provided by third parties, that enable access to web data sources. Web data services are usually discovered according to different features, related to lightweight descriptions. Recent approaches in literature convey on new research challenges, considering also collective intelligence in developers' networks, containing information about service co-usage in existing applications and ratings on services given by developers who used them in their own development experiences. Following this direction, in this paper, we contribute with a distinguishing viewpoint, by proposing an *explorative approach*, that enables web applications developers to iteratively discover services of interest by also relying on collective intelligence, in a Web 2.0 context.

Keywords: Web data service model · Exploratory search · Collective intelligence · Web-oriented architecture

1 Introduction

Exploratory search techniques and tools, that enable users to browse and discover information shared over the web, facing the increasing volume and heterogeneity of available data, are attracting more and more interest from the research and industrial community [1]. Building data-intensive web applications more and more requires frameworks to support the discovery of web data services, that enable access to huge repositories of data and must not be developed from scratch (e.g., Google Maps), according to the Web-Oriented Architecture (WOA) style. In this context, data exploration is possible only through exploration of services used to access data.

Nevertheless, service exploration presents distinguishing features compared to data exploration. Firstly, applications are designed through a sequence of service selections, where a selection step starts from other services previously considered for the application that is being developed. Secondly, who is searching for services expects them to be provided in a specific and subjective way. This paves the way to approaches that, beyond functional and non functional

© Springer International Publishing AG 2017
J. Cabot et al. (Eds.): ICWE 2017, LNCS 10360, pp. 378–385, 2017.
DOI: 10.1007/978-3-319-60131-1_23

requirements, use collective intelligence to suggest services based on the service usage experiences of other developers [5–7]. Our contribution in this paper is the proposal of an explorative approach, that is modeled as a sequence of exploration steps between the system, used to select services, and the developer. The exploration takes care of the collective intelligence in developers' networks, containing information about past service usage experiences in terms of service co-occurrence and rating on services by other developers. The explorative approach is based on a multi-perspective model that includes: (i) a *service-base perspective*, focused on descriptions of services to be explored; (ii) a *service-experience perspective*, focused on the relationships between services and developers who used them to build their own applications and rated services according to their experience; (iii) a *service-abstraction perspective*, focused on features used to describe services (e.g., tags, categories); (iv) a *service-collective-intelligence perspective*, that gives an overview on the network of services as used by the community (developers' service usage experience to be exploited for enabling the exploration). The elements considered in the first two perspectives have been already defined in [2]. We add here the other perspectives as a basis to guide data service exploration. Developers are the main beneficiary of the approach, since they are provided with easy data service exploration to build new applications with minimal effort.

The paper is organized as follows. Section 2 provides the background of the approach and its motivations. Section 3 describes the multi-perspective model. In Sect. 4 we present the data service exploration procedure based on the model. Section 5 presents a proposal for an interactive interface for service exploration. Finally, Sect. 6 closes the paper.

2 Background and Motivations

The approach will be focused on web data services. It deals with available service descriptions (as published in public repositories, like `Mashape.com` and `ProgrammableWeb.com`), that are lightweight characterizations through textual descriptions, categories and technical features (such as protocols and data formats to use them). These lightweight descriptions, which relieve service providers from the burden of providing complex descriptions of supplied services, is one of the success factors for RESTful services. On the other hand, they made challenging the study of effective search facilities.

As an example, let's consider a developer who is developing a new application for hotel booking. Let's suppose that the developer is using `ProgrammableWeb.com` and starts by specifying `hotel booking` keywords to search for services. The repository returns 80 services[1], as shown in Fig. 1. The popularity of `Cleartrip Hotel` service (followed by 164 users) and applications (18 mashups) where hotel booking services have been used (highlighted in figure) are not exploited by the system and it is up to the developer to improve his/her search by considering them.

[1] https://www.programmableweb.com/search/hotel%20booking.

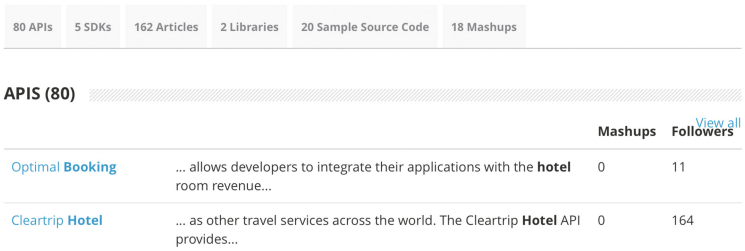

Fig. 1. An example of `hotel booking` service search using `ProgrammableWeb`.

Recent web service recommendation approaches rely on lightweight descriptions of services to overcome the complexity of traditional state-of-the-art approaches on data service discovery (e.g., [8]), that are hampered by the availability of complex, structured service descriptions (e.g., WSDL, WADL and semantic web service formalisms). They exploit categories, tags or semantic tags to search for services, with the application of advanced IR techniques to enhance topic-based service recommendation [3], natural language API description [4], latent factors (e.g., related to the perceived QoS) that affect users to make service selection, identified mainly using matrix factorisation techniques [6]. In this context, several approaches like leverage factors to estimate past experiences of service usage are considered, such as votes/ratings assigned by users to services [7] and the number of times a service has been used in the past and the co-occurrence of services in existing applications [5].

However, these approaches provide the developer with look-up search results, without taking into account decisions made by the developer during the development process by interacting with the search tool. Developers should be progressively supported in refining their requirements. They might not have still taken any decision about the use of additional criteria to search for hotels, such as the number of stars of the hotel (that may have an impact on the final price) as well as the proximity to a specific location. The system may help developers in refining the request, choosing among hotel booking services that accept further constraints such as the number of stars or the location, taking into account the popularity of different solutions and the kind of application that is being developed compared to existing applications where services have been used in the past.

3 Multi-perspective Model for Service Exploration

Starting from the motivations as presented in the previous section, we propose here a multi-perspective model to enable service exploration based on their lightweight descriptions and collective intelligence about their use. The model has inter-perspective relationships as shown in Fig. 2 and the explorative process presented in the next section is based on it.

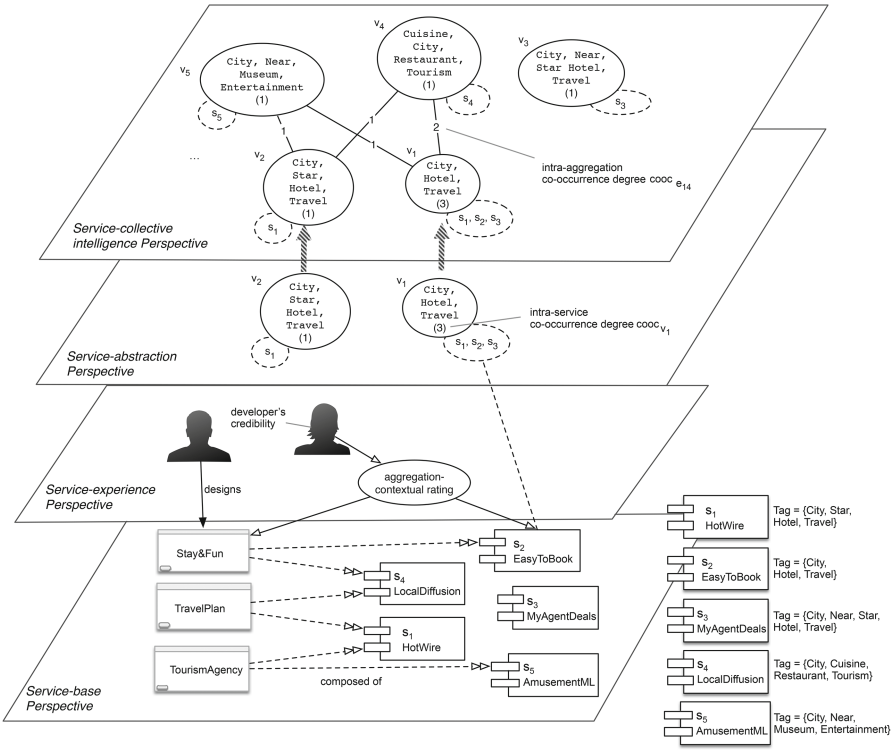

Fig. 2. Overview of the multi-perspective model for exploration purposes.

Service-base perspective. This perspective collects and describes services, aggregations and composition relationships between them. For the purpose of data service exploration, we describe a data service s_i as a set of tags $\mathcal{T}_{s_i}$ used to provide a terminological characterisation of the service (*terminological equipment*), as extracted from its lightweight description. For other service descriptive features, not explicitly mentioned here (e.g., categories, technical features like data formats and protocols), we refer to [2]. An *aggregation* g represents a set of services that will be integrated to deploy a web application. Concerning application development, according to the WOA style, developer has to explore the set of available services, select the most suitable ones, mashup them in the final application. We focus here on service exploration for selection purposes, and we talk about service aggregations, instead of web applications, where the latter ones are the final product of the development process. Figure 2 shows three aggregations, composed of five data services. Service terminological equipments are shown as well.

Service-experience perspective. This perspective is focused on the set $\mathcal{D}$ of developers who designed service aggregations. In this perspective we focus on collective knowledge related to aggregation design experience, modelled around two concepts: (a) the relationship between a service s, an aggregation g and

a developer d, who designed g using s; (b) ratings assigned by developers to a service s when used within an aggregation g (*aggregation-contextual rating*). In [2] we provided more details about aggregation-contextual rating.

Service-abstraction perspective. Services described in the *Service-base perspective* and sharing common terms in their terminological equipments are grouped and abstracted here through a set $\mathcal{V}$ of nodes. In particular, each node $v_i \in \mathcal{V}$ is described as $v_i = \langle \mathcal{T}_{v_i}, cooc_{v_i} \rangle$, where $\mathcal{T}_{v_i}$ is a set of common terminological items (i.e., their intersection) used to describe a number $cooc_{v_i}$ of data services (*intra-service term co-occurrence degree*). In Fig. 2 {`City`, `Hotel`, `Travel`} have been used to describe three data services (namely, s_1, s_2 and s_3).

Service-collective-intelligence perspective. Edges are established between nodes defined in the previous perspective in order to build a *term graph*, which synthesizes the collective intelligence on services. Formally, the graph is represented as $\langle \mathcal{V}, \mathcal{E} \rangle$, where $\mathcal{V}$ is the set of nodes, as previously defined, and $\mathcal{E}$ is the set of edges. Each edge $e_{ij} \in \mathcal{E} \subseteq \mathcal{V} \times \mathcal{V} \times \mathbb{N}$ is formally described as $e_{ij} = \langle v_i, v_j, cooc_{e_{ij}} \rangle$, where services associated with nodes v_i and v_j have been jointly used within $cooc_{e_{ij}}$ aggregations (*intra-aggregation term co-occurrence degree*). In Fig. 2 $cooc_{e_{14}} = 2$ since service s_4 has been used together with services s_1 (`TravelPlan`) and s_2 (`Stay&Fun`). The difference here with respect to approaches that consider co-occurrence of specific service instances (e.g., [5]) is that the intra-aggregation term co-occurrence enables developers to explore services that have not been aggregated yet, but can be considered for aggregation based on their "term similarity" with other services. For example, service s_3 could be suggested to be used together with s_4 because of its term similarity with s_1 and s_2. This will enable a greater coverage of proposed solutions, at the cost of a lower precision, that can be acceptable in an explorative process. The term graph can be built and maintained in a fully automatic way, without the need of human intervention.

4 Data Service Exploration

We envision the service exploration process as a sequence of exploration steps between the developer and the system, used to search for services. The developer starts the exploration by specifying the set $\mathcal{T}^r$ of terms used within the search request, that provide some initial hints about developer's interests. The system suggests services by computing similarity, filtering and ranking techniques such as the ones introduced in [2]. Let's denote with $\mathcal{S}^e$ the set of search results, recommended by the system. The developer can modify the set $\mathcal{T}^r$ to look for new services or can choose some services from the search results $\mathcal{S}^e$ to include them in the work-in-progress aggregation g^r. The system reacts to developer's actions by supporting exploration according to three modalities.

– **Exploration by simple search.** The system also looks for nodes $v_i \in \mathcal{V}$ such that $\mathcal{T}^r \subseteq \mathcal{T}_{v_i}$. If multiple nodes are found, for each $v_i \in \mathcal{V}$ the system will suggest to the developer additional terms to be included within the set $\mathcal{T}^r$ considering the set $\mathcal{T}_{v_i} \setminus \mathcal{T}^r$. A suggestion is given for each $v_i \in \mathcal{V}$, ranked in

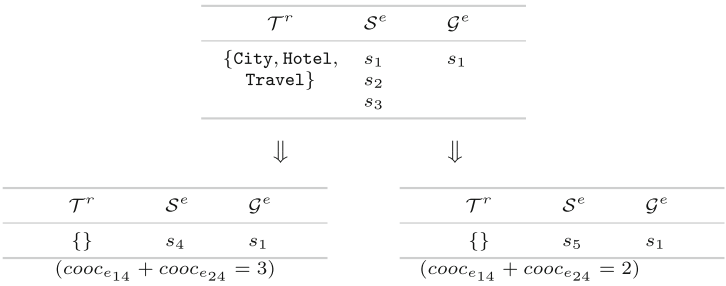

Fig. 3. Example of exploration by proactive completion.

decreasing order with respect to the $cooc_{v_i}$ value. The developer can explore these suggestions in order to consider services alternative to $\mathcal{S}^e$ and to formulate a different request. For instance, if $\mathcal{T}^r = \{\texttt{City}, \texttt{Hotel}, \texttt{Travel}\}$, the system might also suggests as additional terminological item the term $\{\texttt{Star}\}$ first ($cooc_{v_2} = 2$), and $\{\texttt{Near}\}$ as second option ($cooc_{v_3} = 1$). In this way, the developer might realize that hotels can be searched either based on the number of stars or based on the proximity to a given location and he/she might refine the request by choosing one of the two options.

– **Exploration by proactive completion.** The developer selects a subset $\overline{\mathcal{S}^e} \subseteq \mathcal{S}^e$ of services he/she is interested in. The system suggests services that could be used together with services in g^r, by updating the set $\mathcal{S}^e$, according to the *intra-aggregation co-occurrence*. Let's consider the example shown in Fig. 3. After performing a search based on $\mathcal{T}^r = \{\texttt{City}, \texttt{Hotel}, \texttt{Travel}\}$, thus obtaining $\mathcal{S}^e = \{s_1, s_2, s_3\}$ as results, the developer chooses s_1 to be included in g^r. With reference to Fig. 2, s_1 is associated with v_1 and v_2 nodes. Considering node v_1, other nodes connected to v_1 by graph edges are v_4 (associated with s_4, $cooc_{e_{14}} = 2$) and v_5 (associated with s_5, $cooc_{e_{15}} = 1$). Similarly, considering node v_2, $cooc_{e_{24}} = 1$ and $cooc_{e_{25}} = 1$. Therefore, the system ranks better the service s_4 than s_5, since $cooc_{e_{14}} + cooc_{e_{24}} > cooc_{e_{15}} + cooc_{e_{25}}$. The developer can accept one of these results. If more than one service is included in g^r, the step of retrieving services is repeated for each service in g^r.

– **Exploration by hybrid completion.** This explorative modality is a combination of proactive completion and simple search. After $\mathcal{S}^e$ has been updated, the developer selects a subset $\overline{\mathcal{S}^e} \subseteq \mathcal{S}^e$ of services he/she is interested in, as well as he/she specifies a new set $\mathcal{T}^r$ of terms. The system suggests services that could be used together with services in g^r, by updating the set $\mathcal{S}^e$. In order to obtain this set, a proactive completion step on g^r retrieves some services as explained before.

5 Considerations About the Exploration Interface

The interface of a tool aimed to support explorative search should better serve result set examination and item comparison, being conceived as a combination of

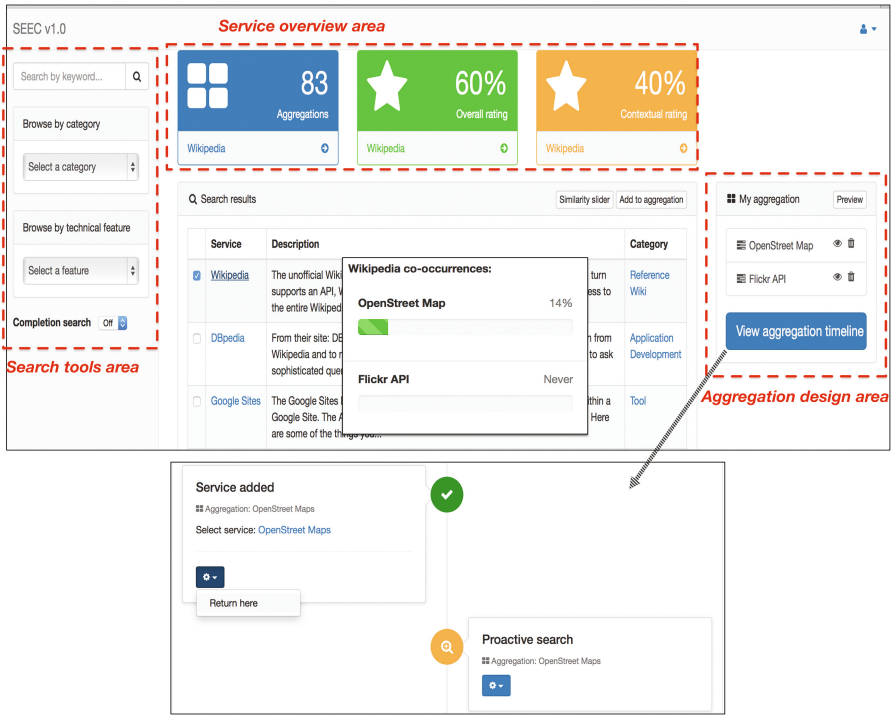

Fig. 4. Data service exploration web interface (with example of exploration timeline).

browsing and analytical strategies. Figure 4 depicts the web interface for exploratory search of services. Faceted metadata (e.g., categories, technical features) provide an effective entry point for exploration and selection. Therefore, search and browsing facilities are provided to enable developers to search services by keywords, and browse available services by category or technical features (`Search tools area`). A flag also enables to activate completion search (either proactive or hybrid). When the developer moves the mouse on one of the search results, the number of uses of service and its rates are shown within the `Service overview area`, as well as co-occurrences of pointed service with other services in the aggregation that is being developed are shown in a popup window. This gives an immediate view of the search results suitability; the developer can move among search results according to overview information and perform multiple lookup searches, then he/she can select a service and include it into the aggregation that is being developed using the `Add to aggregation` button. By clicking on the `View aggregation timeline` button (on the right), the developer can browse in a timeline the history of his/her exploration steps, where for each step the status of the developed aggregation in that moment, the kind of search (simple, proactive or hybrid) and search criteria used in that step are shown. It is possible for the developer to return to any exploration step by selecting the `Return here` option on the step.

6 Concluding Remarks

In this paper, we proposed an approach for data service explorative search, based on collective intelligence in developers' networks, containing information about past service usage experiences in terms of service co-occurrence and ratings on services by other users. The approach is built on a sequence of steps, where the user receives suggestions from the system based on the past interactions. Future work will be devoted to the study of techniques for including latent factors (e.g., related to the perceived QoS) in the exploration process. Further open research challenges concern integration within the system, which implements the approach, of a query engine for multi-source data access and usability experiments on the visualization interface to further enhance the exploration experience of developers. This will move the approach towards non-expert users, who will be able to access the web of data without specific web application development skill.

References

1. Idreos, S., Papaemmanouil, O., Chaudhuri, S.: Overview of data exploration techniques. In: ACM Conference on Management of Data (SIGMOD) (2015)
2. Bianchini, D., Antonellis, V., Melchiori, M.: Capitalizing the designers' experience for improving web API selection. In: Meersman, R., Panetto, H., Dillon, T., Missikoff, M., Liu, L., Pastor, O., Cuzzocrea, A., Sellis, T. (eds.) OTM 2014. LNCS, vol. 8841, pp. 364–381. Springer, Heidelberg (2014). doi:10.1007/978-3-662-45563-0_21
3. Cao, B., Liu, X., Li, B., Liu, J., Tang, M., Zhang, T.: Mashup service clustering based on an integration of service content and network via exploiting a two-level topic model. In: Proceedings of the 23rd International Conference on Web Services (ICWS 2016) (2016)
4. Xiong, W., Wu, Z., Li, B., Gu, Q., Yuan, L., Hang, B.: Inferring service recommendation from natural language api description. In: Proceedings of the 23rd International Conference on Web Services (ICWS 2016) (2016)
5. Gao, W., Chen, L., Wu, J., Bouguettaya, A.: Joint modeling users, services, mashups and topics for service recommendation. In: Proceedings of the 23rd International Conference on Web Services (ICWS 2016) (2016)
6. Liu, X., Fulia, I.: Incorporating user, topic, and service related latent factors into web service recommendation. In: IEEE International Conference on Web Services (2015)
7. Balakrishnan, R., Kambhampati, S., Manishkumar, J.: Assessing relevance and trust of the deep web sources and results based on inter-source agreement. ACM Trans. Web **7**(2), 32 (2013)
8. Vaculin, R., Neruda, C., Sycara, K.: Modeling and discovery of data providing services. In: Proceedings of the 2008 IEEE International Conference on Web Services, pp. 1032–1039 (2008)