# Semantic Resource Framework

Marco Brambilla, Alessandro Campi, Stefano Ceri, and Silvia Quarteroni

Politecnico di Milano, Dipartimento di Elettronica e Informazione
Piazza L. Da Vinci, 32. 20133 Milano, Italy
{mbrambil,campi,ceri,quarteroni}@elet.polimi.it

**Abstract.** The Semantic Resource Framework (SRF) is a multi-level description of the data sources for search computing applications. It responds to the need of having a structured representation of search services, amenable to service exploration, selection, and invocation. The SRF aims at extending the Service Mart model used so far in search computing to overcome some of its limitations. The main new features include *external attributes*, which represent the input to be provided by users for accessing objects; *selector attributes*, describing the possibility to map the same access pattern to different services based on some condition; *key attributes* for objects; and a generalized notion of *nearness* between objects. The high-level view presented by SRF is a very simple Entity-Relationship model with objects and binary connections, that can be used for very different query tasks, ranging from custom search applications (i.e. predefined queries) to exploratory search (i.e., exploration of its objects and connections) to natural language interfaces (i.e., query dialogues). Such high-level view should be considered as an initial step in the enrichment of the service repository with additional semantic capabilities.

**Keywords:** service repository, ontology, semantic annotation, service description, search services.

## 1 Introduction

Service registration is an essential aspect of the Search computing project; the process is very critical because it must satisfy two conflicting requirements. On one side, services must be described with enough details about their interfaces and deployment so as to support their composition and invocation by means of fully automatic processes. On the other side, the actual mapping of services to real-world objects and facts must be exposed, so as to enable the construction of high-level user interfaces covering the semantic gap between user interaction and service selection.

The service model used for registration must describe not only the object or fact exposed by a service, but also the logic that a specific service performs while accessing an object, so that an interpretation system can select the specific service which best matches the user's requirements expressed in an informal or semi-formal way. Moreover, the model must support processes that aim at recognizing, at service registration time, when services describe the "same" objects or properties through "different" notations (e.g., names or types), so as to support matching. The scope of service registration in SeCo is therefore quite broad, as it must cover aspects ranging

S. Ceri and M. Brambilla (Eds.): Search Computing II, LNCS 6585, pp. 73–84, 2011.

from performance indicators up to the semantic description of services and of their parameters.

The Service Mart model adopted so far in SeCo for service description [6] uses a multi-level modeling approach, consisting of conceptual, logical, and physical layers. The *conceptual level* is a very simple model which characterizes real world entities, called *Service Marts (SM)*, structurally defined by means of attributes, and their relationships. The *logical level* describes the access to the conceptual entities in terms of data retrieval patterns (called *Access Patterns, AP*) described by input and output attributes. Finally, the *physical level* represents the mappings of these patterns to concrete Web *Service Interfaces (SI)*, which incorporate the details about the endpoint and the protocol to be used for the invocation, together with some basic statistics on the service behavior. These in turn may be used for granting agreed levels of quality of service (QoS).

The motivation for a three-layered architecture is due to the following needs: (1) abstracting the conceptual properties of objects from the large amount of services that access them; (2) applying the separation of concerns principle to the service description task, by granting the independence of concept definitions, access methods, and concrete service descriptions.

In particular, the Service Mart model accomplishes the abstraction objective by forcing schema uniformity throughout layers; in this way, it supports a clear definition of how services can be composed and joined. Unfortunately, schema uniformity between layers and service implementations does not always correspond to real-world situations. For instance, the support of object access according to different criteria may require different augmentations of the schema, so as to include additional attributes which describe the access to objects in different context of use. The purpose of this paper is to extend the Service Mart model by making it more expressive and more fitting to the search service description requirements.

The extensions introduced in this paper include: *external attributes*, which represent the input to be provided by users for accessing objects and as such need not to be mapped to object properties; *selector attributes*, describing the possibility to map the same access pattern to different services based on a condition; the definition of *key attributes* for objects; and, for certain domains (such as geographic location), the notion of *nearness* that can substitute for equality in the join of two services.

SRF is a first step in order to add more semantics to service registration; in the next two chapters of the book, further semantic extensions are studied, consisting in the annotation of services with ontological knowledge [3] and in the direct use of an ontological model substituting the SRF [20]. These extensions are under consideration in the project for future extensions.

Section 2 presents a state-of-the-art of the literature in the field. Section 3 briefly summarizes Service Marts from [6] so as to make this paper self-contained. Then, Section 4 introduces innovative aspects of the model (external, selector, and key attributes), and Sections 5 introduces “nearness”, i.e. the possibility of accessing objects or of connecting pairs of objects ranked by their “nearness” to other objects, which is defined for given properties (e.g., spatial location, time, money). Section 6 presents a high-level view of SRF that conforms to the Entity-Relationship model, and can be seen as a first step towards adding semantics to service descriptions.

## 2 Related Work

The work described in this paper is the result of a research stream starting with [19], where the authors propose a Web service management system that enables querying multiple Web services in a transparent and integrated fashion and propose an algorithm for arranging a query's Web service calls into a pipelined execution plan that exploits parallelism among Web services. In this context, SRF is a proposal for increasing the abstraction level and thus facilitating the choice of services and the definition of plans. In the following, we revise the proposals that address similar issues.

Our **conceptual level** description of services through service marts is in line with [8], which describes the idea of the Web of concepts. In this work the term concept refers to things of interest to users of the Web who are either searching for information or trying to accomplish some task. The shift from a Web of pages to a Web of objects is now a recognized trend [2] and several mainstream search engines are following the line by introducing new features in this direction. In [7] the authors propose a conceptual model that describes actors, activities and entities involved in a service-oriented scenario through a glossary of terms.

Also the **logical level** several approaches exist for describing objects on the Web. The most popular ones are based on Google Fusion Tables [11][13][14], a cloud-based service for data management and integration. Fusion Tables enables users to upload tabular data files and provides ways of visualizing the data (e.g., charts, maps, and timelines) and the ability to filter and aggregate the data. It supports the integration of data from multiple sources by performing joins across tables that may belong to different users. There are also several projects related to structured data at Google. Google Public Data[1] is an effort to import public government data and provide high-quality and carefully chosen visualizations of data in response to search queries. The Google Squared Service[2] lets users specify categories of objects and explore attributes of these entity sets. In this case, the data populating the tables is automatically extracted from various sources on the Web, and may not always be accurate.

At the **physical level**, the trend toward the Web of objects is well represented by the Linked Data initiative, which has recently seen an increasing amount of shared information, also thanks to initiatives like the W3C Linked Open Data (LOD) community project[3] and to the dedication of prominent Semantic Web researchers[4]. Outside this initiative, the major search engines are providing facilities for accessing information sources through APIs and query languages. The most known resource is YQL (Yahoo! Query Language)[5] is a language and a platform that lets Web applications query and filter data from different sources across the Internet through SQL-like statements. Similarly, Google Base API[6] also allows one to upload structured data and to query it through the Web. Another relevant aspect of concrete service description is the specification of relevant information for quality of service (QoS) support. QoS has been thoroughly studied

[1] http://www.google.com/publicdata/home
[2] http://www.google.com/squared
[3] http://esw.w3.org/SweoIG/TaskForces/CommunityProjects/LinkingOpenData/
[4] http://www4.wiwiss.fu-berlin.de/bizer/pub/LinkedDataTutorial/
[5] http://developer.yahoo.com/yql/
[6] http://code.google.com/apis/base/

in the past (see e.g. [15], [17]). In line with these approaches, we keep track of service performance for dynamic runtime optimization of execution plans.

Finally, several proposals, such as DAML-S [1], OWL-S [16], COSMO [18] and WSMF [9][10], extend the description of services in a semantic direction, as we plan to do in future work (see the next chapters [3][20] for an overview of the possible research directions).

## 3 Service Marts

Service Marts are specific data patterns; their regular organization helps structuring Search Computing applications. The most well known data modelling pattern is the so-called "data mart" used in the context of data warehouses; a data mart is a simple schema having one core entity, describing facts, surrounded by multiple entities, describing the dimensions of data analysis [5]. It facilitates the expression of operations for data selection and aggregation (e.g. data cubes, rollup, drilldown). Analogously, a "Web mart" [11] is a pattern introduced in the Web design community to characterize the role played by data items in data-intensive Web applications. Service Marts are instrumental in supporting the notion of "Web of objects" [10] that is gaining popularity as a new way to think of the Web, going beyond the unstructured organization of Web pages.

Figure 1 provides an overview of the approach, by presenting a sample concept Movie, registered as a service mart, together with two associated access patterns and service interfaces, with the respective attribute mappings (notice that mappings are shown only for the first access pattern for clarity). The next subsections describe the semantics and notation of each level.

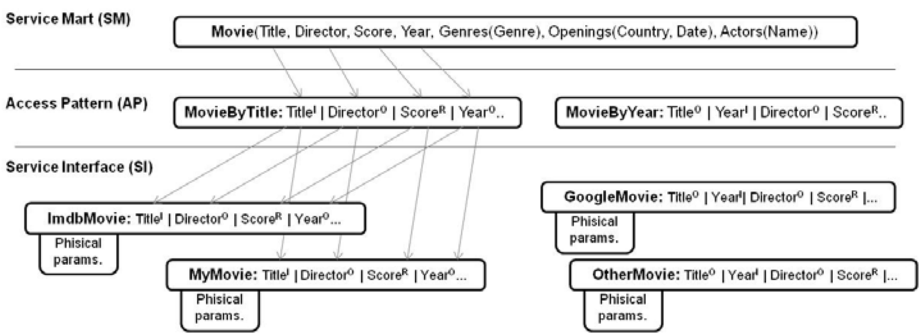


**Fig. 1.** Example of descriptions for accessing Movie information through the Service Mart, Access Pattern, and Service Inteface layers

### 3.1 Conceptual Level

At the conceptual level, the definition of a Service Mart includes the object's name and the collection of the object's attributes. All attributes are typed; attributes can be atomic (single valued) or part of a repeating group (multi-valued); each repeating group is a non-empty set of attributes that collectively defines a property of the Service Mart with multiple values (such as "genres" or "actors" of a given movie). The model choice is to support structural complexity with only one level of nesting, rather than arbitrary nesting. The conceptual description of the object "Movie" is:

```
Movie(Title, Director, Year, Score, Language, Genres(Genre), Actors(Name, Sex))
```

## 3.2 Logical Level

At the logical level, Service Marts are associated with one or more access patterns representing the signatures of service calls. Access patterns contain a subset of the Service Mart's attributes tagged with I (input), O (output), or R (ranking). Ranking attributes may be visible in output. For ease of understanding, if the service call is mapped to a parametric query, input attributes provide query parameters, output attributes provide results, and ranking attributes are used for ordering result instances.

$\texttt{Movie}_1\texttt{(Title}^O\texttt{, Director}^O\texttt{, Score}^{RO}\texttt{, Year}^O\texttt{, Language}^I\texttt{, Genres.Genre}^I\texttt{, Actors.Name}^O\texttt{)}$

$\texttt{Movie}_2\texttt{(Title}^I\texttt{, Director}^O\texttt{, Language}^O\texttt{, Genres.Genre}^O\texttt{, Actors.Name}^O\texttt{, Actors.Sex}^O\texttt{)}$

$Movie_1$ accesses movies by Language and Genre (i.e., "action movies in English"), and results are ranked by Score (a new attribute). $Movie_2$ accesses movies by matching their Title to a string (e.g. "Ben Hur"). We expect few (zero, one, more) un-ranked results.

## 3.3 Physical Level

At the physical level, each access pattern may be mapped to several service implementations. Each implementation is characterized by a physical URI to be called, a set of physical properties that are specific to the implementation, and a mapping between logical attributes and physical parameters. Services are divided into *exact services* (producing a set of equivalent responses) and *ranked services* (search systems producing a list of results ordered by priority, sometimes explicitly ranked, typically not exhausted by the calling process).

## 3.4 Connection Patterns

Connection patterns are high-level abstractions of "real world relationships" that provide a simple interface to users and hide implementation details. They are built by means of attributes that share the same domains. At the conceptual level, they are defined by a non-directed edge with a name, e.g.:

```
PlayingMovie(Movie,Theatre)
```

At the logical level they are defined by a (possibly directed) edge with name and join condition:

```
PlayingMovie(Movie,Theatre): (Title=Movie.Title)
```

The above edge represents a join operation between the two access patterns and can either be directed or undirected. In the first case, information is "piped" from one access pattern to another, along connection attributes which are in output in the first service and in input in the second service. As an example, the connection between the following access patterns is directed, following the information flow from $Movie_1.Title^O$ to $Theatre_1.M.Title^I$:

$Movie_1(Title^O, Director^O, Score^{RO}, Year^O, Language^O, Genres.Genre^I, Actors.Name^O)$

$Theatre_1(Name^O, Address^O, Movie.Title^I, Movie.StartTime^O)$

Undirected edges are present when join attributes are both in output from the services, as shown by the following example; note that the Title attribute is labelled "O" in both patterns.

$Movie_1(Title^O, Director^O, Score^{RO}, Year^O, Language^O, Genres.Genre^I, Actors.Name^O)$

$Theatre_2(Name^O, Address^O, Movie.Title^O, Movie.StartTime^O)$

# 4 Model Extensions

After experimentation, we realized that the basic Service Mart model of the world encountered a number of limitations, and therefore some extensions were needed. We start by extending the model with three orthogonal features: external, selector, and key attributes. Each such feature is described next.

## 4.1 External Attributes

External attributes are "new" attributes appearing at the logical level, while not being present at the conceptual level; they support object access and ranking. External attributes are appended to access patterns, as illustrated by Figure 2. Attributes for object access support multi-value and similarity access:

- The former case occurs when a query includes one or more values of a multi-valued attribute (e.g., "internet access", "parking", and "wellness area" as required properties of a hotel); in such a case, the service call signature requires providing one or more values in input, which the invoked service will use to select object instances (the specific access predicate being used by the service, e.g. "at least two" required properties for a hotel, is not of our concern).
- The latter occurs when a query finds the instances that most closely match a given input (e.g. the movies whose title most closely matches the word "apocalypse"); in such a case, the service call signature requires two attributes for the same domain, one given in input and one extracted as output, and ranking of selected object instances is based upon best matching.

External attributes for ranking allow services to explicitly describe how their result instances are ranked. They may correspond to specific object attributes (e.g. "stars" or "scores" for hotels) which provide an explicit ranking, or just represent the "position" of the result instance in the result list, used by services returning opaque ranking.

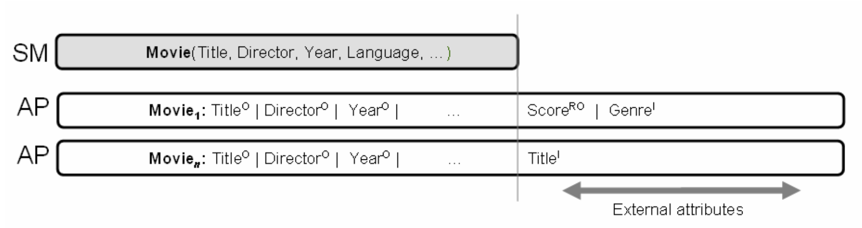


**Fig. 2.** External attributes for access patterns

### 4.2 Selector Attributes

Selector attributes support the selection of specific service implementations; they may be present at the conceptual level or be added at the logical level; in the latter case, selector attributes must be labelled "I" (for input). Figure 3 shows "language" as a selector attribute, present in the Service Mart used for accessing *n* movie services (e.g. one for movies in "English" and one for movies in "Italian").

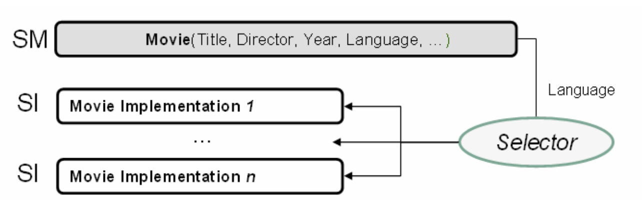


**Fig. 3.** Selector attributes to choose a specific service implementation

### 4.3 Key Attributes

In the context of the SeCo project, object identity is used for supporting object caching (so as to avoid repeated queries to the same service) and query sessions (so as to progressively build information relative to the "same" real world object). Both these uses capitalize upon the specific choice of service implementation, performed by the query compiler, and therefore properly belong to the physical level. The latter is also the only level where object identity can be maintained, as it is not realistic that object identity be maintained across different service implementations accessing distinct sets of objects. Thus, we allow each service implementation to be optionally associated with a local definition of "key attributes", corresponding to one or more attributes collectively forming the (primary) identifier of the object instances retrieved by the service implementation. Such key attributes may be present at the conceptual or logical levels or may be added at the physical level. For example, depending on the service interface, hotels may be identified by the pair "city, number" or by an "OID" or "URI" generated by the service.

# 5 Nearness Support in Accessing Resources

In the Service Mart model, connections of access patterns are based on value equality (equi-joins). However, queries often call for value similarity and/or partial matching; in this case, knowledge of the underlying semantics of attributes supports nearness computation. This aspect is indeed crucial in Web search and intrinsically leads to the concept of ranking based on join conditions. Furthermore, it provides a fundamental rationale for the introduction of external attributes.

In SeCo, the most relevant domain for similarity support is the *spatial domain,* which is used for geographic locations such as the "addresses" of resources. Indeed, the spatial domain is the most common example of application of nearness support as many types of queries are naturally geo-localized, i.e. their results can be visualized upon maps. The trend of offering geo-localized results is common to most search engines, such as Google or Yahoo!

Further domains providing value similarity are the *temporal domain* (describing dates and times) and the *economic domain* (describing costs). Similarity computation between *lexical strings* is also performed to compute term relatedness; this is achieved via morphological operations such as stemming or via external vocabularies such as WordNet [21]. String comparisons may be refined for specific application domains (e.g., the Bio-SeCo application using GeneOntology [12], described in Part 7 of this book). Next, we illustrate similarity support in the spatial domain and focus on providing results close to the current user's location.

(a)

| GoogleMovies | AP: Theatre1 | URI: http://... | | | |
|---|---|---|---|---|---|
| TTL=6000, chunksize=10, cacheable=true, ***provides=Spatial Near*** | | | | | |
| UserAddress | Name | Address | M.Title | M.StartTime | ... |
| IAddr | Name | OAddr | MovieTit | MovieTime | ... |

(b)

| SpatialNear | System | URI: http://... |
|---|---|---|
| TTL=600, chunksize=1, cacheable=1, ... | | |
| Input1, Input2: GeoCoordinates | Output: Distance (Km) | |

**Fig. 4.** Service interface for (a) a service natively providing the "Spatial Near" function; (b) an ad-hoc service providing spatial nearness calculation

## 5.1 Nearness in Single Resources

Ranking by nearness is supported natively by many search services. For instance, the "GoogleMovies" service (shown in Fig. 4(a)) outputs movie shows ranked by distance from the location given as input[7]. In this case, it is sufficient to label the service with the supported similarity semantics, a feature recognized by the SRF.

[7] See e.g. http://www.google.it/movies?near=washington%square%new%york

Note that in Fig. 4(a), "UserAddress" (input location) is modelled as an *external attribute* of the $\text{Theatre}_1$ access pattern.

In addition, generic services exist which return the distance between two addresses, provided either as geographic coordinates or as <country, city, street> triples. Such services may be used to rank resources based on their distance from a specific location. Fig. 4(b) represents an ad-hoc service, supported within the SeCo query engine, which takes two coordinates as input and produces their distance as output.

Services for supporting value similarity are engineered starting from services supporting value equality; a number of candidate locations, produced e.g. by location-aware resource selection services, are used to feed the first input parameter of the ad-hoc service, while the second input parameter is set to the user's current location. A sorter is then used to order candidate locations by distance. Caching of triples representing two locations and their distance can be used to reduce calls to the ad-hoc service, and the Search Computing engine supports execution strategies that limit the size of the result set to avoid waiting too long for sorted results (note that sorting is a blocking operation, i.e. an operation that can only be executed when its full input is available).

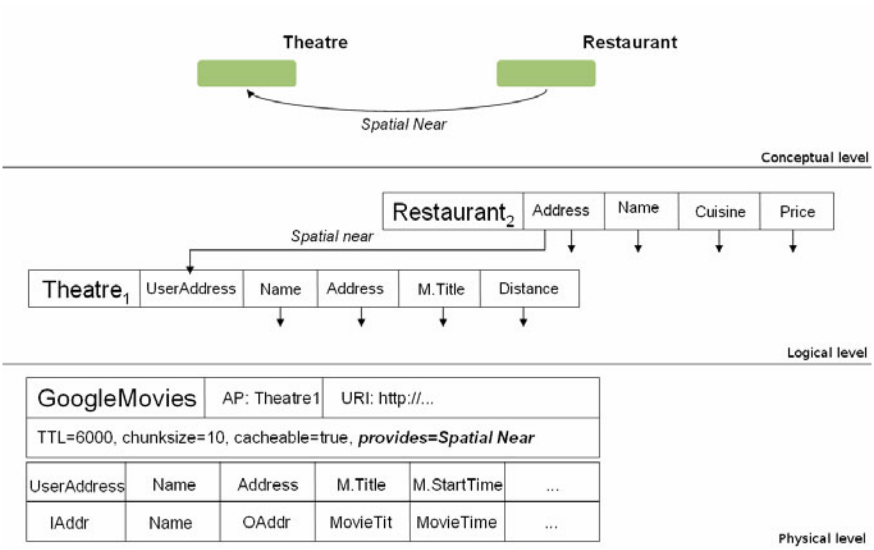


**Fig. 5.** GoogleMovies service providing spatial nearness: modelling at the physical, logical and conceptual levels

## 5.2 Accessing Pairs of Resources

The approach illustrated in Section 6.1 can be extended to connection patterns. Fig. 5 shows an example of use of the GoogleMovies service for supporting geographic nearness as a connection pattern. The query searches for theatres (and their movies) close to restaurants, selected (and ranked) in turn according to the user's preference. This is possible thanks to the match between external attributes "Address" and

"UserAddress" in the $Restaurant_2$ and $Theatre_1$ access patterns. Note that the connection at the conceptual level is now directed (because the nearness function is supported by the "theatre" service) and labelled with the name of the *SpatialNear* function.

Alternatively, an ad-hoc service can be used to support geographic nearness, by using the scheme illustrated in Fig. 6. The ad-hoc "SpatialNear" service can only be invoked once the services for "restaurants" and "theatres" have been called, as they use address pairs as input. Note that in this case the connection at the conceptual level, labelled with the name of the *SpatialNear* function, is not directed.

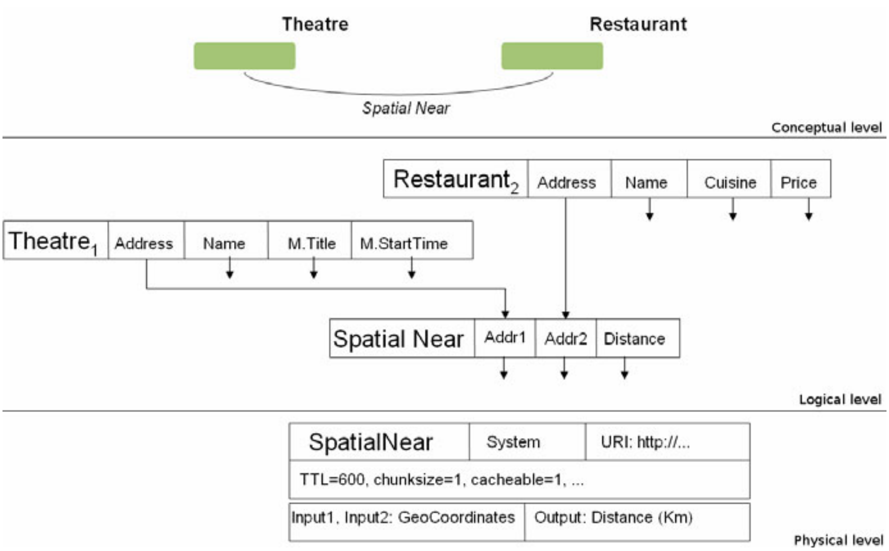


**Fig. 6.** SpatialNear ad-hoc service: modelling at the physical, logical and conceptual levels

## 6 Top-Level View of the Semantic Resource Framework

While the service mart model is ideal for registering individual services, as it clusters several service descriptions within a hierarchical multi-level view, its evolution into the SRF highlights fact that services collectively describe a given "domain of discourse", i.e. a particular subset of reality which can be the target of SeCo queries.

The top-level view of SRF is a simple Entity-Relationship model, as described in Figure 7; it defines the application context, characterized by the presence of named entities (service marts) and relationships (connection patterns). Among the possible semantic meanings of relationships, of course we rightfully include nearness as discussed in Section 5.2. This view abstracts away from the complexity of mapping service interfaces to data sources and of integrating the different names and formats used by each source to represent its properties, and focuses on a simple, semantic view.

Thanks to this view, the exploration of information and the definition of search queries is simplified and made more efficient. The "focus" of the exploration can start with a single object and then progressively add more objects, thereby building queries with a fashion, as discussed in the first chapter of this book. Future work for supporting keyword-based or natural language interfaces will also be based on high-level representations of the universe of discourse. Thus, this semantic view, although very simple at this stage, is an important step in the direction of adding semantic power to service descriptions.

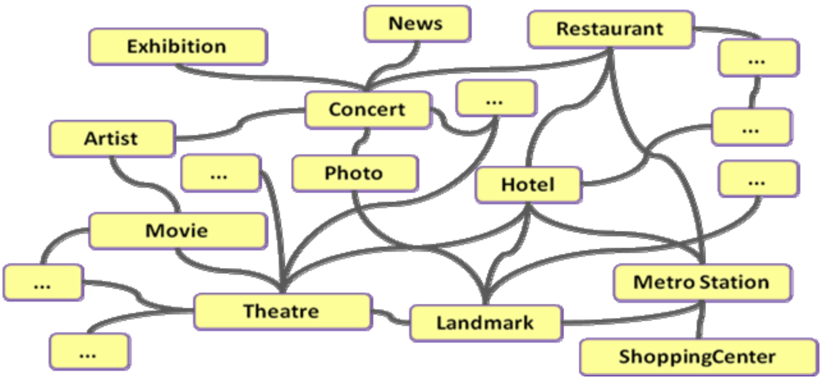


**Fig. 7.** Example of high-level view offered by SRF

## 7 Conclusions

Search service management is one of the most critical aspects in search computing. In this chapter, we discussed our current approach to search service conceptualization and registration; the model extends our previous approach in many important directions, such as the support of external, selector, and identifier attributes, and the support of nearness for specific semantic domains, such as distance and time. Registration tools, described in [5], support service registration according to this model and subsequently let query designers specify the search query upon it.

We envision progressive extensions of the SRF for incorporating more semantics; such extensions concern the use of semantic annotations, and the use of ontological knowledge at the conceptual level of the model, thereby adding semantics to the current Entity-Relationship description. These extensions, discussed in the next two chapters [3][20], will further empower designers at registration time, and will facilitate the support of high-level queries.

## References

[1] Ankolenkar, A., et al.: DAML-S: Web service description for the Semantic Web, `http://www.daml.org/services/daml-s/2001/10/daml-s.html`

[2] Baeza-Yates, R., Raghavan, P.: Next Generation Web Search. In: Ceri, S., Brambilla, M. (eds.) Search Computing. LNCS, vol. 5950, pp. 11–23. Springer, Heidelberg (2010), doi:10.1007/978-3-642-12310-8_2

[3] Bergamaschi, S., Beneventano, D., Po, L., Sorrentino, S.: Automatic Schema Mapping Through Normalization and Annotation. In: Ceri, S., Brambilla, M. (eds.) Search Computing II. LNCS, vol. 6585, pp. 85–100. Springer, Heidelberg (2011)

[4] Braga, D., Ceri, S., Daniel, F., Martinenghi, D.: Optimization of multi-domain queries on the Web. In: Proc. VLDB, vol. 1(1), pp. 562–573 (August 2008)
[5] Brambilla, M., Tettamanti, L.: Search computing processes and tools. In: Ceri, S., Brambilla, M. (eds.) Search Computing II. LNCS, vol. 6585, pp. 169–181. Springer, Heidelberg (2011)
[6] Campi, A., Ceri, S., Maesani, A., Ronchi, S.: Designing service marts for engineering search computing applications. In: Benatallah, B., Casati, F., Kappel, G., Rossi, G. (eds.) ICWE 2010. LNCS, vol. 6189, pp. 50–65. Springer, Heidelberg (2010)
[7] Chakrabarti, K., Ganti, V., Han, J., Xin, D.: Ranking objects based on relationships. In: Proc. of SIGMOD Int. Conf. on Management of Data, New York, USA, pp. 371–382 (2006)
[8] Dalvi, N., et al.: A Web of Concepts. In: PODS 2009, Providence, Rhode Island, USA, June 29-July 2 (2009)
[9] Fensel, D., Bussler, C.: The Web Service Modeling Framework WSMF. Electronic Commerce Research and Applications 1(2), 113–137
[10] Fensel, D., Musen, M.: Special Issue on Semantic Web Technology. IEEE Intelligent Systems (IEEE IS) 16(2)
[11] Google Fusion Tables, http://tables.googlelabs.com/
[12] Gene Ontology, http://www.geneontology.org/
[13] Gonzalez, H., Halevy, A., Jensen, C., Langen, A., Madhavan, J., Shapley, R., Shen, W., Goldberg-Kidon, J.: Google fusion tables: Web-centered data management and collaboration. In: Proceedings of the 2010 International Conference on Management of Data, SIGMOD 2010, Indianapolis, USA, June 06-10, pp. 175–180 (2010)
[14] Gonzalez, H., Halevy, A., Jensen, C., Langen, A., Madhavan, J., Shapley, R., Shen, W.: Google Fusion Tables: Data Management, Integration, and Collaboration in the Cloud. In: Proceedings of the ACM Symposium on Cloud Computing, SOCC (2010)
[15] Liangzhao, Z., Benatallah, B., Ngu, A.H.H., Dumas, M., Kalagnanam, J., Chang, H.: QoS-aware middleware for Web services composition. IEEE Transactions on Software Engineering 30(5), 311–327 (2004)
[16] Martin, D., Burstein, M., et al.: Bringing Semantics to Web Services with OWL-S. In: World Wide Web, vol. 10(3), pp. 243–277 (September 2007)
[17] Maximilien, E.M., Singh, M.P.: A framework and ontology for dynamic Web services selection. IEEE Internet Computing 8(5), 84–93 (2004)
[18] Quartel, D.S., Steen, M.W., Pokraev, S., Sinderen, M.J.: COSMO:A conceptual framework for service modeling and refinement. Information Systems Frontiers 9(2-3), 225–244 (2007)
[19] Srivastava, U., Munagala, K., Widom, J., Motwani, R.: Query optimization over Web services. In: VLDB 2006, VLDB Endowment, pp. 355–366 (2006)
[20] Suchanek, F., Bozzon, A., Della Valle, E., Campi, A., Ronchi, S.: Towards an Ontological Representation in Search Computing. In: Ceri, S., Brambilla, M. (eds.) Search Computing II. LNCS, vol. 6585, pp. 101–112. Springer, Heidelberg (2011)
[21] WordNet, http://wordnet.princeton.edu/
[22] Xi, W., Fox, E.A., Fan, W., Zhang, B., Chen, Z., Yan, J., Zhuang, D.: SimFusion: measuring similarity using unified relationship matrix. In: Proc. of the 28th Int. ACM SIGIR Conf. on Research and Development in Information Retrieval, New York, USA, pp. 130–137 (2005)
[23] Zaragoza, H., Rode, H., Mika, P., Atserias, J., Ciaramita, M., Attardi, G.: Ranking very many typed entities on wikipedia. In: CIKM 2007: Proc. of the 16th ACM Conf. on Information and Knowledge Management, pp. 118–1018. ACM, New York (2007)