Automatic Web Service Composition Using Web Connectivity Analysis Techniques

W3C Workshop on Frameworks for Semantics in Web Services 2005 Position Paper

John Gekas and Maria Fasli
{jgekas, mfasli}@essex.ac.uk,
Computer Science Department,
University of Essex, UK.

1. Introduction - Position

Web service integration in Service-Oriented Architectures (SOA) has been an active research area lately, as a result of the increasing impact the web services deployment paradigm on today's World Wide Web. Indeed, many authors have argued that this will be the dominating paradigm on the web in the years to come [CASATI *et al*., 2001], thus changing its structure from a web of information and manually-accessible applications, to that of a web of automatic services.

A number of approaches addressing the problem of automatic web service integration have been presented recently, both academic and industrial. To name a few, [PEER, 2003] suggests the use of existing AI planning systems, even though he does not present a solution prototype; [TUT and EDMOND, 2002] on the other hand, propose the use of patterns in service composition. According to this approach, each composition request is matched against an appropriate abstract *composition template*, which is mapped to existing services at the implementation level. [MONTEBELLO and ABELA, 2002], [KOPENA and REGLI, 2003] and [PONNEKANTI and FOX, 2002] follow a rule-based appoach, proposing the use of expert systems (JTP – Java Theorem Prover and JESS – Java Expert System Shell).

We argue that the majority of the approaches presented in the area, however succesful and applicable they may be, fail to take into consideration two important factors inherent to service composition: the dynamics of data flows among web service operations, and the potentially dynamic structure of the service composition search space. We believe that taking into account these factors can provide us with powerful heuristics in order to guide the composition process. In this paper, we attempt to give an overview of our methodology for automatic web service composition: we regard service composition as a search problem, where the search space consists of all the potential web service operations that can be part of a workflow. Our approach focuses on automatically extracting information about the *connectivity structure* of a semantic web service reqpository (the *registry*) and using this information in order to guide the search process.

To illustrate our methodology, we focus on a typical service composition use case: both the provided input parameters and the required output parameters are explicitly defined in the composition request, and solution can be given with a sequential execution of web service operations, not dependent to each other in any way. More business-oriented scenarios (such as Supply Chain Management, for instance), along with issues like workflow control constructs, composition-wide constraints, service dependencies and selection mechanisms based on QoS (Quality of Service) criteria are addressed by our work, but description of the corresponding mechanisms is beyond the scope of this paper. Before we describe our service composition methodology in more detail, we give a brief overview of our framework's architecture.

2. Framework Architecture

Our framework consists of a repository-style registry, where web service semantic descriptions are stored. We have to note here, that no actual web services are hosted at the registry; however, semantic descriptions of already existing services are stored; this allows web services from various providers to be integrated within our system. We also have to add that our research has been focused on two particular kinds of web services: informational services (such as a financial news service) and E-Commerce services (e.g. Amazon transaction service). This choice was deliberate: Web-based

statistical observations led us to believe that these are the dominating types of services populating today's web. The overall architecture of our system is illustrated in Fig.1:

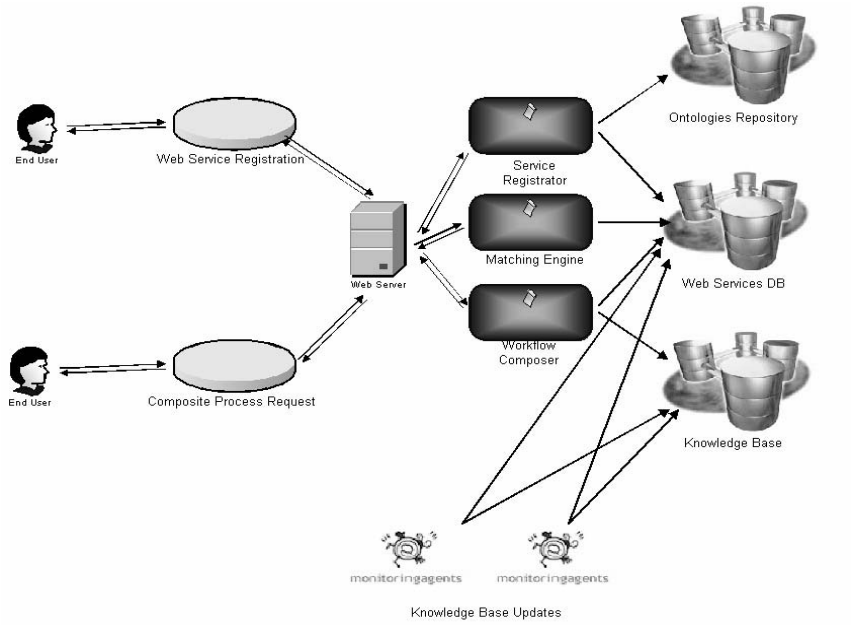

Fig 1: Framework Architecture

Following is a brief description of our system's components:

- **Web service registration system and semantic descriptions storage**: semantic descriptions about each web service are automatically generated based on some information given by the service's provider, and stored using the DAML-S format.
- **Ontologies**: The semantic descriptions generated for each registered service are based on the common ontologies used by the system, which serve as a common vocabulary used by the descriptions. Our ontologies are developed in-house, using the DAML+OIL format. Concepts such as service categories (e.g. InformationService) and semantic data types (SDT's - e.g. ProductName) are defined within the ontologies.
- **Request Matching Engine**: Before any attempt for workflow generation takes place, a given request is matched against the web services registered with our system, in order to check whether a service exists that can satisfy the request.
- **Knowledge Base**: The knowledge base (KB) holds information about the *connectivity structure* of the search space, which serves as guiding heuristic for the composition search process (more details in the next section).
- **Workflow Composer**: the actual algorithm performing composition upon request. Our platform is not bound to one particular algorithm – instead, it is our intention to test our methodology with different algorithm types and examine their performance. The experimental work so far has been performed a depth-first recursive search algorithm.

3. Composition Methodology

In order to guide the composition process efficiently, we are using heuristics derived from the repository of semantic descriptions, regarding the connectivity structure of the repository and how "tightly" various types of web services are linked together (the *ecology* of the search space). Connectivity structure figures are estimated automatically by the system and updated as new services are added. Alternatively, the above figures could be calculated offline by a search space crawler – this method however could be computationally too demanding for large search spaces (service registries). In this sense, we have used a similar methodology to the one used by major search engines in order to rank individual web resources and estimate the connectivity graph of sub-parts of the web.

Various approaches [PAGE and BRIN, 1998], [KLEINBERG, 1997] describe ways to rank individual web resources based on their popularity. This way, a major-scale search engine can reason on how "important" a web resource is, in terms of how many other resources point to it and how many

resources it points to itself. Even though most approaches follow a similar pattern, we could say there are two broad categories: the first estimates the overall "importance" of a web resource, whereas the second one estimates the importance of a web resource relatively to a particular area/request.

We believe that some aspects of search engines' ranking methodology can be applied to our problem domain: Google (http://www.google.com) founders [PAGE and BRIN, 1998] describe their algorithm, titled PageRank (henceforth PR) as follows: *"…given a web page A and a web page B, the PageRank value of A is the probability that a 'random web surfer' will visit page B just by following links initiated from A*". Following a similar train of thought, we can also define the probability that a certain web service B can be reached at a certain depth, given that another web service A has been accessed. Web services can also be considered as web resources that point to other web resources. Assume a web service A that provides a semantic data type (SDT) X as an output parameter, and a web service B that accepts data type X as its input parameter. In this case, we assume that there is a forward link between A and B. Likewise, we can define backward links as well.

Furthermore, we could define similar connectivity measures for semantic data types themselves: a semantic data type that is being used by X% of the search space as an Input parameter, is considered to have a forward PR equal to X%, whereas a data type that is being used by Y% of the operations belonging to our search space as output parameter, is thought to have a backward PR equal to Y%.

Based on the above information, the composition process works as follows: At every stage of the search process, the algorithm examines the PageRank value of the corresponding operations that are linked to the current node, and decides which node to follow next based on which one is more "important" – the other possible nodes are placed at a "Priority Queue" based on their PR values. Using an iterative PageRank for higher depths would probably give us a better idea about how important a web service is, but cannot be calculated at composition time since it is computationally demanding, and we have been trying to keep offline processing to a minimum.

4. Experimental Work

We have performed some initial experiments in order to evaluate the performance of our approach. For the whole of SDT's that are present in the InformationService category, we have generated every single combination among them, on a 1-on-1 basis. For each of these combinations, we have generated an appropriate request and run our algorithm against all the requests.

Some initial experiments have shown that our methodology performs better when the PageRank figures of the semantic data types of the InformationService category is distributed analogously at increasing depths: otherwise, a SDT with a relatively high PR at low depths, which drops to a much lower PR at a higher depth is possible to lead the search process to a dead-end. Furthermore, we have to note that PageRank, by its nature, is request-independent: it is estimated as an overall percentage of the whole search space – as a side effect, it can be expected that nodes with a relatively high PR can "attract" the search algorithm, even though the particular nodes may not be related to the specific composition request.

5. Conclusions – Further Work

We have presented a methodology for automatic web service composition that views the composition search space as a dynamic hyperlinked environment of scalable size, and uses the environment's link connectivity structure in order to guide the composition process effectively. Our purpose is to take advantage of the scalable and dynamic structure of the composition search space, and the data flow dynamics at composition time. Based on some initial experimentations, we have seen in which cases this methodology can work effectively. Furthermore, to our knowledge, no other service composition framework following the above principles has been proposed at the time of writing.

On the other hand, we believe that our methodology presents some weaknesses as well - mainly, as mentioned above, the fact that PageRank is request-independent by its nature. However, PageRank alternatives have been suggested by search engine technology researchers, which rank web resources in accordance to specific request areas: for instance, [KLEINBERG, 1997] and the HITS algorithm. In fact, some existing search engines use the HITS methodology (such as http://www.teoma.com). As a

main further work direction, we are planning to test our methodology after implementing a request-sensitive ranking system, following this logic.

6. References

[CASATI *et al*., 2001] Fabio Casati, Ming-Chien Shan and D. Georgakopoulos (2001). "E-Services - Guest editorial." The VLDB Journal 10(1): 1.

[KLEINBERG, 1997] J. Kleinberg. Authoritative sources in a hyperlinked environment. Journal of the ACM, 1999. Also appears as IBM Research Report RJ 10076, May 1997. http://www.cs.cornell.edu/home/kleinber/auth.ps.

[KOPENA and REGLI, 2003] DAMLJessKB: A Tool For Reasoning With The Semantic Web, *2nd International Semantic Web Conference (ISWC2003)*, October 2003, Florida, USA.

[MONTEBELLO and ABELA, 2002] DAML Enabled Web Services and Agents in the Semantic Web. *NODe 2002 Web and Database-Related Workshops on the Web*, p.46-58, 2002.

[PAGE and BRIN, 1998] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. In WWW Conference, volume 7, 1998. http://www7.scu.edu.au/programme/fullpapers/1921/com1921.htm

[PEER, 2003] Peer, Joachim: Towards Automatic Web Service Composition Using AI planning Techniques (first draft), 2003.

[PONNEKANTI and FOX, 2002] SWORD: A Developer Toolkit for Web Service Composition, *WWW2002, THE ELEVENTH INTERNATIONAL WORLD WIDE WEB CONFERENCE*, Honolulu, Hawaii, USA, 7-11 May 2002.

[TUT and EDMOND, 2002] The use of Patterns in Service Composition. *Revised Papers from the International Workshop on Web Services, E-Business, and the Semantic Web*, 2002, p.28-40.