Discovery and Composition of Semantic Web Services

Ms.S. Jegatha,
ME Computer Science and Engineering,
Mookambigai College of Engineering,
Pudukkottai-622502
jegatha.hcc@gmail.com

Mr.N.Nagapaandian, M.E
Assistant Professor,
Mookambigai College of Engineering,
Pudukkottai-622502

*Abstract***— Flexible and dynamic interoperation of autonomous information system often requires automated combination of several highly distributed and heterogeneous web services.web service composition aims to accelerate rapid application development and service reuse as well as to provide access to a variety of complex services. There are several approaches developed for web service composition. Graph Based Technique has been taken, which allows the users and applications to discover, deploy, compose and synthesize services automatically. The goal of semantic web service composition is to provide machine -readable descriptions of the web services. WSDL Language used for describing the Web Services in terms of their,input, output, precondition, effects and their process model. The salient feature of this approach is its optimality and scalability, i.e., its ability to handle very large service repositories, and it's extremely efficient processing times for discovery and composition queries as well as provides fine-grained input/output interface.**

*Index Terms***—Semantic, Web Services , Clustering, Graph Based Technique.**

I. INTRODUCTION

Web Services make up a connection technology. It is a way to connect services together into a service-oriented architecture. Primary elements of Web Services are Repository, Messaging, and Services. Web Services refers to the technologies that allow for making connections. Services are what you connect together using Web Services. A service is the endpoint of a connection. The technology of Web Service is the most likely connection technology of service-oriented architectures. Web service is a program accessible over the web that may affect some action or change in the world (i.e., causes a side-effect). Examples of such side-effects include a web base being updated because of a plane reservation made over the Internet, a device being controlled, etc. An important future milestone in the Web's evolution is making services ubiquitously available. As automation increases, these Web services will be accessed directly by the applications rather than by humans. In this context, a Web service can be regarded as a "programmatic interface" that makes application to application communication possible. An infrastructure that allows users to discover, deploy, synthesize and compose services automatically is needed in order to make Web services more practical.

1.1 Need Of Semantic Service Discovery And Composition

To make services ubiquitously available we need a semantics-based approach such that applications can reason about a service's capability to a level of detail that permits their discovery, deployment, composition and synthesis. Informally, a service is characterized by its input parameters, the outputs it produces, and the side-effect(s) it may cause. The input parameter may be further subject to some pre-conditions, and likewise, the outputs produced may have to satisfy certain post-conditions. For discovery, composition, etc., one could take the syntactic approach in which the services being sought in response to a query simply have their inputs syntactically match those of the query, or, alternatively, one could take the semantic approach in which the semantics of inputs and outputs, as well as a semantic description of the side-effect is considered in the matching process.

1.2 Requirements for Web Service Composition

Traditional composition approaches require human-intensive involvement, making them time-consuming and error prone. Therefore, the ability to automatically or semi-automatically orchestrate web services in a short timeframe is highly desirable. Generally Web Service Composition has the following Phases..

Specification Phase

In this phase, the user specifies the goal he is trying to achieve from creating a complex task to be achieved through the composition of web services and produces an abstract specification. The specification should have enough detail to aid in creating the abstract specification and include functional and non-functional requirements. Functional requirements are constraints, control/data flow and high-level goal of the complex task.Non-functional aspects may include security policy, QoS constraints, etc. Desired behavioral issues (i.e. termination conditions, failure recovery requirements) should also be able to be specified.

Planning Phase

The planning phase takes the abstract specification created in the specification phase and produces an abstract composite workflow. If the specification did not contain enough detail (i.e., it was a partial specification) or was stated as P/E and goals, then the description needs to be decomposed into simpler steps in this phase.

Validation Phase

The validation phase takes an abstract composite workflow that is an output of the planning phase to address the following:

• Syntactic validation: Is the workflow well-formed and structurally correct (i.e. does not contain deadlocks, infinite cycles, etc)? Syntactic validation may be handled by the planning tool, depending on the approach used.
• Semantic validation: Does the plan satisfy user goals and requirements/constraints that were detailed in the specification phase?
If the user created more than one abstract composite workflow in the planning phase, there may be multiple candidates to validate. The result of the validation should produce a syntactically and semantically optimal abstract composite workflow.

Discovery Phase

The discovery phase takes the abstract composite workflow, and finds suitable atomic services for each task in the workflow by querying service repositories.

Execution and Monitoring Phase

The execution and monitoring phase provides deployment and execution of a newly created composite service. This phase includes following control flows that were specified in the workflow and recovery mechanisms to ensure proper execution of composition.
Within Each phase the requirements needed for web service composition:
• Specification phase: Provide an easy way for a user to specify task goals, requirements and constraints without extensive domain knowledge.
• Planning phase: Provide an automatic way to compose an abstract workflow based on the specification.
• Validation phase: Provide techniques to ensure that the composite process realized via an abstract workflow satisfies the user's stated task goals.
• Discovery phase: Provide a way to discover services that satisfy task specifications in the workflow.
• Execution with monitoring phase: Provide a framework for monitoring and executing service, and provide automatic fault- handling mechanisms.

II. RELATED STUDIES

M. Carman, L. Serafini, and P. Traverso, (2003) 'Web Service Composition as Planning,' in ICAPS 2003 Workshop on planning for web services, pp. 1636–1642.

Web service composition problem can be viewed as a planning problem in which state descriptions are ambiguous and operator definitions are incomplete. Discusses the problem of interpreting documents (which describe the world state), and introduces a semantic type matching algorithm. The matching algorithm together with an interleaved search and execution algorithm allow for basic automated service composition.

Approach is based on data type matching and interleaved search and execution for service composition. Type matching algorithm based on the idea that the target type needs to be shown to be a more general version of the source. In this type matching algorithm,when we compare the goal (target) type tgoal , to a particular service output (source) type tout , we require that tgoal subset *of* M tout , which is to say that all documents conforming to the output type also conform (form a subset of those conforming) to the goal type after a certain mapping *M* has been applied to them

- Services that are described in a standard and possibly formal manner, i.e. all services which provide the same functionality are called in the same way, require the same inputs and produce the same outputs.
- Difficulties associated with the heterogeneity of the web services planning domain, and are able to apply techniques from "simpler" (or at least more homogeneous) domains such as database query processing.

P. Hennig and W.T. Balke, (2010) 'Highly Scalable Web Service Composition Using Binary Tree-Based Parallelization,' IEEE Int. Conf. on Web Services, pp. 123–130.

Data intensive applications, e.g. in life sciences, pose new efficiency challenges to the service composition problem. Since today computing power is mainly increased by multiplication of CPU cores, algorithms have to be redesigned to benefit from this evolution. Framework is developed for parallelizing service composition algorithms investigating how to partition the composition problem into multiple parallel threads. But in contrast to intuition, the straightforward parallelization techniques do not lead to superior performance as our baseline evaluation reveals. To harness the full power of multi-core architectures, proposes two novel approaches to evenly distribute the workload in a sophisticated fashion

Web service composition frameworks need a representation model for compositions. Recently, binary trees have shaped up as the most efficient method. Building on these tree structures, framework for parallelization has been designed which is to be self-contained. The main idea is to create a binary tree for each composition request and store all information of the current composition process in this data structure.

- To detect and prevent composition loops by more advanced techniques like computing hash values of composition paths instead of just restricting the binary tree depth.
- Loops are formed while compositing services which plays the major problem as well as there no methodology for searching the binary tree in a depth manner.

O.Hatzi, D. Vrakas, M. Nikolaidou,(2011) 'An Integrated Approach to Automatic Semantic Web Service Composition through Planning', Transactions on Service Computing,pp1-14

It describes the integrated approach for automatic semantic web composition through planning. An important advantage of the composition is the composition process, as well as the discovery of atomic services that take part in the composition, are significantly facilitated by incorporating of semantic information. OWL-S web service descriptions are transformed into planning problem described in a standardized fashion using PDDL.

The proposed approach is based on transforming the web service composition problem into a planning problem and solving it after enrichment with semantic information extracted from OWL-S. In this way, extensive research in AI planning can be applied to the area of web service composition. The produced domain is described using well-established standards, such as PDDL, while solutions may be acquired using a variety of external planners in a standard way. Independence from planning techniques and algorithms is provided, enabling us to take advantage of recent research advances. Solutions are transformed back to OWL-S descriptions, which are suitable for execution in any web service environment. The approach facilitates the composition process, even for non-expert users.

- Future goals include the addition of the OWL-S descriptions of produced composite services in the registry of available services, to explore the possibility to accelerate the composition process.
- Moreover, it lies in our immediate plans to study ways to enhance the approach with the ability to produce various composite services according to non-functional user preferences, dealing with pragmatic knowledge.

M. Klusch, A. Gerber, and M. Schmidt, (2005) 'Semantic Web Service Composition Planning with OWLS-Xplan,' in Proceedings of the AAAI Fall Symposium on Semantic Web and Agents.

Present an OWL-S service composition planner, called OWLS-Xplan, that allows for fast and flexible composition of OWL-S services in the semantic Web. OWLS-Xplan converts OWL-S 1.1 services to equivalent problem and domain descriptions that are specified in the planning domain description language PDDL 2.1, and invokes an efficient AI planner Xplan to generate a service composition plan sequence that satisfies a given goal. Xplan extends an action based Fast Forward-planner with a HTN planning and re-planning component.

OWLS-Xplan consists of several modules for pre processing and planning. It takes a set of available OWL-S services, a domain description consisting of relevant OWL ontologies and a planning query as input, and returns a plan sequence of composed services that satisfies the query goal. Xplan is a heuristic hybrid search planner based on the FF-planner. It combines guided local search with graph planning, and a simple form of hierarchical task networks to produce a plan sequence of actions that solves a given problem.

- During plan execution, the agent has to check for each action of the plan whether its preconditions hold, or not.
- If at least one precondition is not satisfied, Xplan gets informed about which facts are invalid, at which position in the plan this problem occurs, and then checks whether the original plan still can be executed.
- Otherwise, it tries to fix the problem by searching for an alternative path in the connectivity graph from the actual position in the plan to the goal state. In addition, it may temporally block unnecessary actions to reduce the search space, thereby avoiding a complete preprocessing phase.

S. Oh, D. Lee, and S. Kumara, (2007) 'Web service planner (WSPR): an effective and scalable web service composition algorithm,' Int. Journal of Web Services Research, vol. 4, no. 1, pp. 1–22.

As the emergence of service-oriented architecture provides a major boost for e-commerce agility, the number of available Web services is rapidly increasing. However, when there are a large number of Web services available and no single Web service satisfies the given request, one has compose multiple Web services to fulfill the goal. Toward this problem, presents an AI planning based Web service composition algorithm named as WSPR.

WSPR activates two-step search, First, it computes the cost of achieving individual parameters staring from *ri to ro* by conducting the forward search; second, it approximates the optimal sequence of Web services that connects *ri* to *ro* by conducting regression search leveraging on the results obtained from the first step as guidance.

- Running time of WSPR spent is more in the forward reasoning stage due to the sheer number of Web services to visit.
- Thus, in order to improve the overall speed of WSPR better ways to be more informed about parameter space are needed in that. It is in still in the formative stage as far as publicly available Web services are concerned, and, it is not found out that there are no compositions with more than two Web services linked.
- Their results points out two lessons: first, composition is in the formative stage as far as publicly available Web services are concerned.
- Second, it is not easy to discover correlations between Web services even while using the semantic description matching. easy to discover correlations between Web services even while using the semantic description matching.

Rao and X. Su,(2004) 'A Survey of Automated Web Service Composition Methods," in Semantic Web Services and Web Process Composition', vol. 3387, pp. 43 54.

In today's Web, Web services are created and updated on the fly. It's already beyond the human ability to analysis them and generates the composition plan manually. A number of approaches have been proposed to tackle that problem. Most of them are inspired by the researches in cross-enterprise workflow and AI planning. This paper gives an overview of recent research efforts of automatic Web service composition both from the workflow and AI planning research community.

AI planning problem can be described as a five tuple <S, S0,G,A, i>, where S is the set of all possible states of the world, S0 S denotes the initial state of the world, G S denotes the goal state of the world the planning system attempts to reach, A is the set of actions the planner can perform in attempting to change one state to another state in the world, and the translation relation S × A × S defines the precondition and effects for the execution of each action.g Workflow is a platform for the specification, enactment and management of composite services. And uses a static workflow generation method. A composite service is modelled by a graph that defines the order of execution among the nodes in the process. The graph is created manually but it can be updated dynamically.

- Because the web service environment is highly complex it is not feasible to generate everything in an automated way.
- Usually the highly automated methods are suitable for generating the implementation skeletons that can be refined into formal descriptions.

B. Srivastava and J. Koehler, (2003) 'Web Service Composition – Current Solutions and Open Problems,' in ICAPS workshop on Planning for Web Services, pp. 28 35.

Composition of Web services has received much interest to support business-to-business or enterprise application integration. On the one side, the business world has developed a number of XML-based standards to formalize the specification of Web services, their flow composition and execution. This approach is primarily syntactical: Web service interfaces are like remote procedure call and the interaction protocols are manually written. On the other side, the Semantic Web community focuses on reasoning about web resources by explicitly declaring their preconditions and effects with terms precisely defined in ontology. For the composition of Web services, they draw on the goal-oriented inter referencing from planning. discusses what makes the Web service composition so special and derive challenges for the AI planning community.

A realistic application domain for Web services (specically, trip planning) and highlight to what extent business needs are addressed by the WSDL Web services and the Semantic Web approaches. And investigating how these approaches differ with respect to the modelling, verification, and deployment of services and the respective inference methods and runtime support that they assume. For composition approach Industry Plan and Artificial Planning has been discussed.

- The Semantic Web provides a process level description of the service which, in addition to functional information, models the preconditions and post conditions of the process so that the evolution of the domain can be logically inferred.
- It relies on ontology's to formalize domain concepts which are shared among services as well as modelling flow composition problem occurs.

[8]Y. Yan, B.Xu, and Z.Gu,(2008) ' Automatic Service Composition Using AND/OR Graph ,' in 10th Conference of E-Commerce Technology,pp.335-338.

As SOC and Web service technology become more widely used, large amounts of services need to be efficiently and effectively composed to meet complex businesses. This paper, proposed an approach to resolve the composition problem over large-scale services. used an inverted table as index for a quick service discovery, and applied a Service Dependency Graph (SDG) and an AND/OR graph as the algorithm basis for parallel composition. Considering the semantic information described in Web service, this approach also recognizes and transmits the semantic relationships described in Web Ontolosgy Language (OWL).

A service discovery and composition model based on the AND/OR graph to resolve the semantic composition. The main idea is to construct an AND/OR graph form a service dependency graph (SDG), applying a bottom-up search algorithm REV* to find a sub-graph for the solution. It is a effective method to find executions on parallel, but it didn't consider the scale of the services. Besides, there have been many research on AND/OR graph searching such as AO* algorithm. And presented a fast service composition model using a inverted table to index the services, also proposed a method to handle the semantic relationship between I/O data.

- The functionalities of services are not considered in the composition method. Obviously, this will cause confusion when two services take and generate exactly the same types of data.
- There is not a benchmark for evaluating a composed service and the composition itself that is widely accepted. For our algorithm, how to balance getting the best solution against getting all the solutions also depend on future experiments.

III. PROPOSED SYSTEM

Composition framework has the following characteristics provide convenient fine-grained discovery mechanisms that could help to discover services able to consume or produce (a subset of) certain types of data as usually required during composition,

Improve the response time of service discovery to process requests very fast,

Support the integration of third party service registries as a key activity in the composition phase,

Incorporate optimizations to improve the scalability of the overall composition process,

Find optimal service compositions by minimizing different criteria such as the number of services or the length of the composition to avoid complex, unmanageable solutions. A graph-based framework focused on the semantic input-output parameter matching of services' interfaces that efficiently integrates the automatic service composition and semantic service discovery

3.1 GRAPH BASED TECHNIQUE

The Proposed system is graph based framework for automatic web service composition. The process is triggered by a composition request that specifies the user requirements in terms of inputs and the expected outputs, as well as the framework Automatically adjusts a composite service plan by removing the non relevant services or adding the services suggested by the system if

their I/O matches semantically.To achieve fine-grained input/output interface entities within a single web service are written into separate web services. Since each and every entities within a single web service is written into separate web services, scalability of the system also increases.
When the system receives a request, the Service Generator computes a graph with all the semantic relations between the relevant services for the request. A request is basically a set of input concepts, which represent the initial set of available inputs, and a set of output concepts, which are the outputs that the composite service should return.

This graph contains all the known services that could directly or indirectly be invoked given the provided inputs. Once the graph is generated, the next step is to apply different optimizations to reduce the graph size in order to improve the optimal composition search performance .This part of the composition is independent of the discovery phase. All the information required to search for the optimal composition is in the graph, namely, the relevant services and the semantic relations between their inputs and outputs, so there is no need to communicate with the disco0very/matchmaking system

A formal framework that presents a theoretical analysis of graph-based service composition in terms of its dependency with a service discovery and we provide a fine-grained I/O discovery interface which reduces the performance overhead without having to assume the local availability and in-memory preloading of service registries. The framework also includes an optimal composition search algorithm to extract the best composition from the graph minimising the length and the number of services, and different graph optimisations to improve the scalability of the system, which as far as we now are not included in other frameworks.

3.1.2ADVANTAGES

• Semantic Matching functionality,

• Semantic Discovery and composition graph Generation

• Graph-Based Optimizations composition search performance.

• The proposed graph search-based algorithm can be applied to the general service composition problems.

• Different from the previous methods, it can generate all the feasible solutions according to a user's request.

IV. CONCLUSION

Semantic Web Service composition approach aims to enable automatic locations, selection, composition and monitoring of web services. The Presented Graph Based Technique focused on the Semantic Input-Output Parameter Matching of Services' interfaces that efficiently integrates the automatic service Composition and Semantic Service Discovery. Here Composition taking Place by means with Fine-grained Input-Output Service Discovery that enables the generation of a Graph Based Composition Which Contains the Set of Services that are Semantically Relevant for an Input-Output Request and Provides Good Performance, and also minimizes over-head between discovery and composition phases.

V.REFERENCES

[1]M. Carman, L. Serafini, and P. Traverso, (2003) 'Web Service Composition as Planning,' in ICAPS 2003 Workshop on planning for web services, pp. 1636–1642.

[2]P.Hennig and W.T. Balke, (2010) 'Highly Scalable Web Service Composition Using Binary Tree-Based Parallelization,' IEEE Int. Conf. on Web Services, pp. 123–130.

[3]O.Hatzi, D. Vrakas, M. Nikolaidou,(2011) 'An Integrated Approach to Automatic Semantic Web Service Composition through Planning', Transactions on Service Computing,pp1-14

[4]M.Klusch, A. Gerber, and M. Schmidt,(2005) 'Semantic Web Service Composition Planning with OWLS-Xplan,' in Proceedings of the AAAI Fall Symposium on Semantic Web and Agents.

[5]S. Oh, D. Lee, and S. Kumara, (2007) 'Web service planner (WSPR): an effective and scalable web service composition algorithm,' Int. Journal of Web Services Research,vol. 4, no. 1, pp. 1–22.

[6]J. Rao and X. Su,(2004) 'A Survey of Automated Web Service Composition Methods,'in Semantic Web Services and Web Process Composition', vol.3387, pp. 43 54.

[7]B. Srivastava and J. Koehler, (2003) 'Web Service Composition – Current Solutions and Open Problems,' in ICAPS workshop on Planning for Web Services, pp. 28-35.

[8]Y. Yan, B.Xu, and Z.Gu,(2008) 'Automatic Service Composition Using AND/OR Graph ,'in 10th Conference of E-Commerce Technology,pp.335-338.