An Evaluation of Dynamic Web Service Composition Approaches*

Ravi Khadka[1] and Brahmananda Sapkota[2]

[1] Faculty of Electrical Engineering, Mathematics and Computer Science
University of Twente, Enschede, The Netherlands
[2] Information System Group, University of Twente, Enschede, The Netherlands

Abstract. Web Services composition has received much interest from both the academic researchers and industry to support cross-enterprise application integration. Promising research projects and their prototypes are being developed. At the same time the web service environment is getting more dynamic as numerous web services are being published by the service providers in the Internet. To meet the users requirements regarding on-demand delivery of customized services, dynamic web service composition approaches have emerged. But still many compositional issues have to be overcome like dynamic discovery of services, compositional correctness, transactional supports etc. In this paper we discuss some of these issues and then investigate some of the representative dynamic web service composition approaches. We evaluate those approaches on the basis of the issues and present how the future research can benefit by addressing those issues of dynamic web service composition.

1 Introduction

In recent years Web services have received much interest as an emerging technology for Business-to-Business Integration (B2Bi) of heterogeneous applications over the Internet [1]. Many enterprises are transforming their business model in Internet with web services because web services technology enables integration of heterogeneous applications regardless of implementation platforms. The full potential of web services as a means for B2Bi solutions will only be realized when existing services and business processes are able to integrate into a value-added composite service. A composite service [2] is a service developed by aggregating the existing services to realize a new value-added functionality. The aggregating process is called (web) service composition. There are two approaches of web service composition: static and dynamic composition.

In static compositions, the aggregation of the services is done at design time. The composition of all the necessary components is determined, bound together and then deployed. This type of composition is more suitable if the business partners involved in the process are fixed and their offered functionalities or the requirements are unlikely to change in short term. Static web service composition is not flexible in the sense that it is not adaptive to the runtime changes when it is in execution and is too restrictive to unavoidable changes in the service requirements [3]. Dynamic web service composition

* The work is done in the context of DySCoTec Project.

Khadka R. and Sapkota B. (2010).
An Evaluation of Dynamic Web Service Composition Approaches.
In *Proceedings of the 4th International Workshop on Architectures, Concepts and Technologies for Service Oriented Computing*, pages 29-41
DOI: 10.5220/0003045100290041

aims at overcoming the problems which are apparent in static web service composition. A dynamic service composition provides flexibility for modifying, extending and adapting changes at runtime [4].

The web service environment is highly dynamic in nature. The number of web service providers is constantly increasing leading to the availability of new services in daily basis [5]. In such a dynamic environment, realizing dynamic web service composition is not so easy because of the following reasons:

- composition process should discover the appropriate components in a composition that is able to transparently adapt the environmental changes.
- composition process should adapt to the customer requirements with minimal user intervention.
- composition process should have transactional support.
- composition process should guarantee composition correctness.
- composition process should also consider Quality of Service (QoS) properties.

With some of above stated issues of dynamic web service composition, we aim at evaluating available approaches of dynamic web service composition with respect to the issues like transaction support, compositional correctness, and QoS support etc. Based on the evaluation we believe that understanding those issues and their importance will pave way for future research directions in dynamic web service composition. A significant number of evaluation papers have been published in literatures for web service composition approaches. The survey of [5] extensively evaluates the various available web service composition approaches based on classifications and compositional issues. In [1], the paper focuses more on web service composition languages like WS-BPEL with WSDL, OWL-S with Golog/Planning. In [6], the authors evaluate the web service composition frameworks based on the level of automation of the service composition process. [7] discusses Web Service Composition and Execution (WSCE). The framework are categorized as interleaved, Monolithic, Staged, and Template-based service composition and execution. We beleive there is no evaluation paper specifically on dynamic service composition. So, the paper summarizes the currently available dynamic web service composition approaches, evaluate the approach according to the framework and then gives an outlook to essential future research works.

This paper is structured as follows: In Section 2, we define a framework used in the evaluation of the different dynamic service composition approaches. In Section 3, we present some of the most relevant dynamic service composition approaches on the basis of a evaluation framework. In Section 4 we evaluate the approaches against the evaluation framework. Finally, we conclude the paper in Section 5.

2 Evaluation Framework

We propose an evaluation framework for the dynamic service composition approaches that are discussed in Section 3. The evaluation framework includes some of the key composition issues and requirements that are identified based on the analysis of [5] and [8]. We do not guarantee the completeness of all the necessary requirements and issues in this evaluation framework. However, we have included some of the important ones in the evaluation framework.

2.1 Transaction Support

Traditional transaction is based on Atomicity, Consistency, Isolation, and Durability (ACID) properties and implicitly assumes closely coupled environment with short-duration activities. But web services often involve loosely coupled systems with long run transactional activities. Using ACID-based transactions in such long running transactional activities could result in undesirable effects like locking of resources for longer time. So, in web service composition it is necessary to provide a flexible transactional supports for the long-running activities in order to guarantee consistency and reliable execution of composite web services that can support coordination, recovery and compensation [9, 10]. WS-Transaction standard, WS-TXM standards are examples of transactional feature support in web service environment. In this paper, we identify whether the transaction support is available in a composition approach being evaluated.

2.2 Compositional Correctness

An important aspect of web service composition is to maintain the correctness of behavior of the composite service. We consider two ways of exploiting compositional correctness of a service composition to preserve the behavioral properties like deadlock free, liveliness, safety etc. The first way is to design the composition algorithm in such a way that it preserves the behavioral properties. The other way is to formalize the composite service and then verify the behavioral properties formally i.e. using some formalization like pi-calculus, petri-nets e.t.c and verify the properties. We believe the later mechanism can be more useful in those cases where the business goal is already achieved but the workflow still might have deadlocks in the path that are yet to be traversed. Using the later mechanism, we can reason about the preservation of the behavioral properties in all possible paths of the workflow of the composite logic. So, formalizing the compositional logic enables us to verify the behavior of the composite service completely. Recently, a variety of concrete proposals from the formal methods community have emerged in order to verify the correctness of the web service composition which is based on state action models or process models [11]. In this paper, we identify whether the approach supports the verification of compositional correctness. In case if it supports, we also identify by which mechanism does it support.

2.3 QoS Support

QoS is used for expressing non-functional properties like performance, reusability, maintainability, security, reliability and availability [12]. In case of loosely coupled environment of web services, the QoS properties can vary greatly because web services can be provided by various third parties and invoked dynamically over the internet [13]. Therefore service compositions must be QoS-aware in terms of understanding and respecting one another's policies, performance levels, security requirements [14]. To support QoS in dynamic service composition, a web service description language that supports QoS description, QoS estimation approach of the composite web service and QoS monitoring is required [15]. In this paper, we identify whether the QoS support is available or if there is partial support in a composition approach being evaluated.

2.4 Automated Composition

We define automated composition as a process of generating an executable process that communicates and binds with a set of existing web services for web service composition and publish itself as a web service with higher level of functionalities. One of the main aims of web service composition approaches is to achieve automated service composition because it enables faster application development and reuse [8]. In absence of automated web service composition, the end user/agent will have to specify the business goal and manually select the available services after the discovery of services matching the requirements and then compose them. In this paper, we identify whether the automated composition feature is available in a composition approach being evaluated.

2.5 Composition Logic Formulation

We define composition logic as the way how the interactions among the participating services take place in the composition process. Under this requirement we investigate how the control flow and the data flow of the composition mechanism are represented. The classification will be either process-driven or rule-driven or hybrid. In process-driven mechanism, the composition uses process definitions where business logic is expressed as a process model to specify possible interactions among the participating web services [16, 17]. In rule-driven approach, business rules are used as composition logic. A business rule [1] is defined as a statement that defines or constrains some aspect of the business. The main features of rule-driven composition approach are that it is purely declarative, highly adaptive and integrated in a truly service oriented approach to business rule management [18]. In the hybrid approach [19], the composition logic is broken down into core part (business processes) and independently evolving, well modularized business rules. In this paper we investigate if the composition approach being evaluated is process-driven, rule-driven or hybrid approach.

3 Dynamic Service Composition Approaches

This section presents a brief overview of some of the prominent dynamic service composition approaches namely: eFlow [20, 21], METEOR-S [22], WebTransact [23, 24], DynamiCoS [25, 26] and SeGSeC [27]. We choose these approaches on the basis of high references in literatures for dynamic service composition. These approaches are then evaluated based on the framework presented in Section 2. Through this evaluation, we identify which of the features are supported by these approaches.

3.1 eFlow

eFlow [20, 21] is a system that supports the specification, enactment, monitoring and management of composite e-services where the composite e-services are modeled as business processes. E-services and web services share commonalities [28] so we interchangeably use those terms in this paper. eFlow runs on the top of E-services Platforms

[1] www.businessrulesgroup.org

(ESPs), such as HP e-speak or Sun Jini which allow the development, deployment and secure delivery of e-services to business and customers.The eFlow model and the overall system provide a flexible, configurable and an open approach to the service composition [29]. The adaptive and dynamic eFlow process model allows processes to adapt the changes in the running environment, perform necessary service execution according to the need of the customer. E-service composition is modeled 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. The graph may include services (represent the invocation of WS), decisions (specify the alternatives and rules controlling the execution flow) and event nodes (enable service processes to send and receive several types of events). Arcs in the graph denote the execution dependency among the nodes.

eFlow includes the notion of transactional region and supports ACID service-level transaction. A transactional region enforces to maintain service level of atomicity. According to the definition of transaction support in Section 2, only preserving the ACID property of transaction is not sufficient to maintain transaction support in web services environment. Hence, transactional feature is not supported in eFLow. eFlow supports the modification of the process for dynamic service composition, but it can not guarantee the correctness of the output. eFlow does not provide QoS modeling capabilities. Service processes in eFlow are able to transparently adapt to environmental changes and dynamically configure at runtime. However, the limitation of the eFlow is that it needs too much manual participation to concretize the generic service nodes (a node in eFlow that supports dynamic process definition for composite services) at execution phase and it does not support the automatic generation of composition for the generic nodes [30]. So, there is no support for the automatic composition. In eFlow, the composite services are modeled as processes that are enacted by a service process engine [29].So, the composition logic is process-driven and a composite service is described as a process schema that composes other basic or composite services.

3.2 METEOR-S

METEOR for Semantic web services (METEOR-S) is a dynamic web service composition framework developed at the University of Georgia which incorporates workflow management for semantic web services. METEOR-S is the follow-up research of Managing End-To-End OpeRations (METEOR). METEOR-S uses semantics for the complete life-cycle of the semantic web services. Its annotation framework is an approach to add semantics to current industry standards such as WSDL. METEOR-S uses techniques from the semantic Web, semantic Web services and the METEOR project to deal with the problems of semantic Web service description, discovery and composition. The Figure 1 depicts the architecture of METEOR-S which has two parts: front end and back end. The front end of METEOR-S is related with annotation and publication of service specifications. The abstract process designer which is related with dynamic composition is a component present at the back end of the METEOR-S. The composition process is initiated by creating the flow of process using the control flow constructs provided by WS-BPEL. The requirements of each service in the process is represented by specifying the service template, which allow to either specify semantic description of the web services or a binding to a known web services. Then, the process

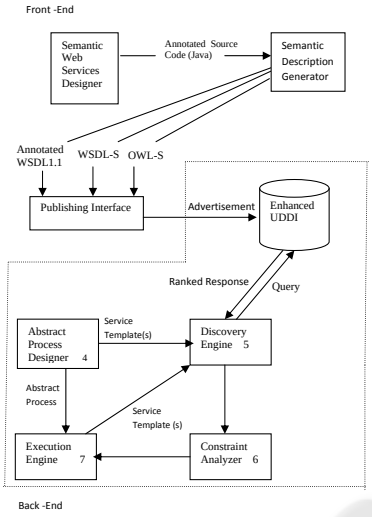

Fig. 1. METEOR-S architecture [22].

constraints for optimization is specified. The details of dynamic service composition in METEOR-S is available on [31].

In METEOR-S architecture, the The constraint analyzer deals with correctness of the process based on QoS constraints. The support for state machine based verification of WS-BPEL process also contributes on the existence of compositional correctness. METEOR-S uses an extensible ontology to represent the generic QoS metrics and domain specific QoS metrics. The cost estimation module of constraint analyzer represents the QoS support. There composition process is not fully automated. The METEOR-S uses process-driven approach for composition. The coordination of the composite service is based on a BPEL-like centralized process engine.

3.3 WebTransact

WebTransact [23] provides necessary infrastructure for building reliable, maintainable, and scalable web service composition. It is composed of a multilayered architecture, an XML-based language named Web Service Transactional Language and a Transaction model. The multi-layered architecture containing a Service Composition Layer, a Service Aggregation Layer, an Integration layer, and a Description Layer are depicted in Figure 2. Based on [24], we provide a brief explanation of web service composition in WebTransact. Application programs interact with the composite mediator services written by composition developers. Such compositions are defined through transaction interaction patterns of mediator services. Mediator services provide a homogenized interface of semantically equivalent remote services. Mediator services also integrate web services providing the necessary mapping information to convert messages from the particular format of the web service to the mediator format.

In WebTransact, an XML-based language named Web Service Transaction Language (WSTL), is used for describing the transaction support. The transaction model

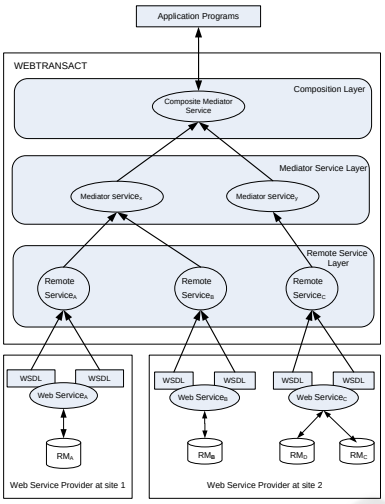

Fig. 2. WebTransact Architecture [24].

of WebTransact provides an adequate level of correctness guarantees when executing the web services composition built with WSTL. Hence, there is transaction support in WebTransact. The transactional model of the WebTransact exploits the dissimilar transaction behavior of web services and guarantees the correct and safe execution of mediator compositions. The notion of correctness of composition execution is based on both the user needs (composition specification) and the *2L-guaranteed-termination criterion* (a weaker notion of atomicity that considers the needs of web service environments). Hence, WebTransact has compositional correctness feature. WebTransact does not provide QoS modeling. The WebTransact approach does not support the dynamic discovery and integration of web services. The Web Services are statically integrated in WebTransact by a developer who plays the role of Web service integrator [24]. So automatic composition is not supported. In WebTransact, web service composition is modeled as composite task by WSTL where a composite task is the combination of atomic task or another composite task. Tasks are identified by its signature, execution dependencies, links and rules. Here rules specify the conditions under which certain event will happen and can be associated with dependencies or to data links and finally evaluating either true or false based on execution [23]. Hence, WebTransact follows the rule-based logic formulation.

3.4 DynamiCoS

In [26, 25], an approach for automated and dynamic service composition named Dynamic Composition of Services (DynamicCoS) is proposed primarily to alleviate the complexity of service composition from the end-users. Upon specifying the service specifications by the users as per their requirements, automatic discovery, matching and the composition of set of services that together fulfill the user's requirement is done by DynamiCoS. The results are then presented to the user who can selects the best suited composition. DynamiCoS represents services in language neutral formalism. A service

is represented as a seven-tuple S =<ID,I,O,P,E,G,NF>, where ID is the service identifier, I is the set of service inputs, O is the set of service outputs, P is the set of service preconditions, E is the set of service effects, G is the set of goals the service realises, NF is the set of service non-functional properties and constraint values. DynamiCoS approach consists of following modules: service creation, service publication, service request, service discovery and service composition. Figure 3 depicts the service composition framework of DynamiCoS. To perform composition, it first organizes the set of services discovered in service discovery phase in a Casual Link Matrix (CLM). The CLM stores all possible semantic connections, or causal links, between the discovered services input and output concepts. Then the graph-based algorithm approach [26] finds a composition of services that fulfil the service request.

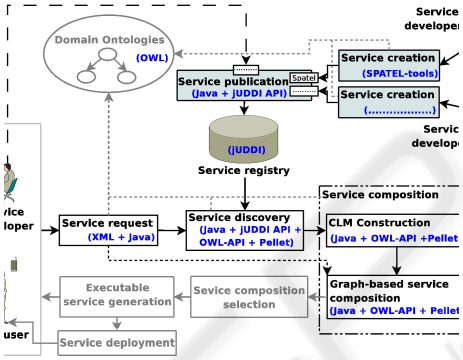

Fig. 3. DynamiCoS Architecture [25].

The main aim of DynamiCoS is to develop dynamic service composition mechanism to support the end-users requirements. Considerable interest is not given in transaction support in DynamiCoS. The service composition module of DynamiCoS builds compositions from service requests and the compositions are correct-by-construction. The algorithm for composition checks for deadlock and also verify if the composition is in accordance with the goals. In DynamiCoS some simple QoS characteristics can be represented and considered in service compositions. Among four ontologies, Non-Functional.owl in DynamiCoS defines non-functional properties for services and hence partially supports QoS modeling. DynamiCoS enbles service creation and publication by service developers at design-time, and automatic service composition by end-users at runtime.The composition is based on semantic graph based composition algorithm, so the composition logic formulation is process-driven.

3.5 SeGSeC

In [27, 32], the authors present a semantic-based dynamic service composition architecture named Semantic Graph-Based Service Composition (SeGSeC) in which the user requests the service in a natural language and the request is then converted into machine-understandable format, i.e. a semantic graph. Based on this semantic graph, SeGSeC composes services. In order to achieve semantic-based dynamic service composition, modeling of the service components and the service composition mechanism

itself must support semantics. To satisfy this requirement, SeGSeC is supported by Component Service Model with Semantics (CoSMoS) and Component Runtime Environment (CoRE). The semantic support in the component modeling is achieved by CoSMoS which integrates the semantic and functional information into semantic graph representation. CoRE functions as middleware and provides functionality to discover and convert different component implementations into a single semantic graph representation. Figure 4 depicts the architecture of SeGSeC.

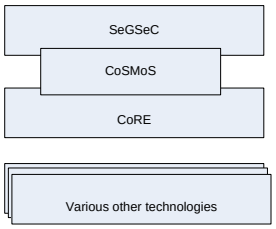

Fig. 4. SeGSeC Architecture [27].

Upon receiving the service request from a user in a natural language, SeGSeC generates the execution path or workflow. The execution path represents the order plan specifying which operations of which component should be accessed in what order. Additionally, SeGSeC also performs the semantic matching to confirm the semantics of the execution plan matches the semantic of the user request.

SeGSeC does not provide transactional support and does not guarantee the compositional correctness. Given the user requirements in the natural language the overall approach generates the workflow such that it satisfies the semantics of the requested services. Starting from the service request from the user to the final composition the process is automated. SeGSeC does not provide QoS modeling capabilities. Upon receiving the user request in the form of semantic graph from CoSMoS, the ServiceComposer component of SeGSeC discovers the components and creates the workflow. In later stage of composition the semantic retrieval rules are applied onto the semantic graph such that the graph models the semantics of the workflow. Hence, SeGSeC has hybrid compositional logic formulation because the workflow is process-driven and later the rules are imposed in composition.

4 Summary and Evaluation

This section presents the overall summary of the comparison and evaluation of the dynamic service composition approaches based on the evaluation framework presented in Section 2. Table 1 presents the summary of the approaches evaluated according to the evaluation framework.

4.1 Transaction Support

Due to the inherent autonomy and dissimilar capabilities of web services, maintaining the transaction support is challenging but still the web service composition requires

Table 1. Summary of the dynamic web service composition approaches.

Approaches	Transaction Support	Compositional correctness	QoS Support	Automated composition	Composition logic formulation
eFlow	No	No	No	No	Process-driven
METEOR-S	No	yes	Yes	No	Process-driven
WebTransact	Yes	Yes	No	No	Rule-driven
DynamiCoS	No	Partial	Partial	Yes	Process-driven
SeGSeC	No	No	No	Yes	Hybrid

an advanced transactional management solution for reliable, consistent and recoverable composition. In this aspect WebTransact approach appears to be better than others since it is supported by a transactional model. The eFlow approach includes the notion of transactional regions and ACID level transactional support which is insufficient in the loosely-coupled web service environment. The remaining three approaches do not ensure the transaction support.

4.2 Compositional Correctness

The compositional correctness ensures the verification of behavior of the composite services. In order to preserve compositional correctness various formal methods techniques can be integrated with the framework. METEOR-S has a support for state machine based verification of WS-BPEL process. However, WebTransact provides another approach to reason about the compositional correctness. It uses compositional specification provided by the user and *2L-guranteed-termination* criterion to verify the compositional correctness [24]. With this feature WebTransact and METEOR-S appear to be the promising one among the others.

4.3 QoS Support

From the evaluation it shows that most of the approaches do not include the QoS properties of the web service composition. Only METEOR-S provides the framework for modeling QoS. There is partial support of QoS modeling in DynamiCoS which is done by by extending the OWL-S. Even the industry based approach eFlow neglects such an important aspect of the composition. In [12], the author mentions two ways to include QoS property namely developing a new language based on semantics like XL and OWL-S and to extend web service description languages like WSDL to support more QoS description. The later one seems to be promising to those approaches where QoS modeling is not included because all of them are based on WSDL.

4.4 Automated Composition

Automated composition enables faster application development without the intervention of the users. From the evaluation and comparison presented earlier, we conclude that use of semantic descriptions or ontology can help in automating the composition process as in DynamiCoS, METEOR-S, and SeGSeC. Including the semantic approach and automating the composition in eFlow can be helpful to the industry.

4.5 Compositional Logic Formulation

Process-driven approach is more suitable for those collaborative businesses where there are few changes in the requirements of business process workflow. It appears to be more effective in terms of managing the workflow because seldom changes are required. But in those collaborative businesses where there are constant changes in the requirements of the business then rule-based systems are advantageous. Hybrid composition approach has its own benefits as it reduces the complexity and avoids monolithic composition [19]. Each part of composition logic in hybrid approach is expressed in the more suitable way either in a business rules or as a process activity providing higher degree of flexibility.

With this evaluation WebTransact and METEOR-S appear to be the most promising approaches. The WebTransact framework is a multidisciplinary work that is related with many other areas such as e-service composition, transactional process coordination, workflow management system and distributed computing systems [24]. However, WebTransact lacks semantic descriptions for automation of the composition process and QoS support. The METEOR-S lacks transactional support and is semi-automated but supports QoS modeling. eFlow is the industrial product and another promising framework but it significantly lacks the transactional support, semantic descriptions for automation and QoS support.

5 Conclusions

Because of the growing trend of transforming existing business models to the Internet with web services and the possibility of creating new web services dynamically, various developments are on-going in the dynamic web service composition. The research and development ranges industry-based products like eFlow to academic researches like WebTransact, DynamiCoS to name a few. In all these developments, many different standards and mechanisms have been proposed but there is still a lack of an overall agreement. In this paper, we presented different dynamic service composition frameworks, ranging from industry based product to the academic research products and compared them in terms of some important requirements and features.

The evaluation shows that transactional support is still missing in most of the approaches which, is one of the most important requirements. So, researches leading to include transactional supports in those approaches and in currently ongoing research approaches will be an important step. The verification of the compositional correctness is also missing in most of the approaches. It is an important aspect however, on the other hand integrating formal techniques in such frameworks is not so easy. In order to verify the correctness, compositional specifications have to be modeled into mathematical formalisms like pi-calculus, process algebra such that the safety and liveliness properties can be explored to detect deadlocks and data consistency in the composition process. The easier way is to include compositional correctness features in the composition algorithm of the framework. QoS support is also the important requirement of the web service composition. In most of the approaches QoS support is not available so research on including QoS modeling is also an open research direction. The researches on the features like especially automated composition of the web service composition

have gained some maturity by using semantics. The use of semantics and ontology in composition has resulted in the automated composition but is still limited in academic researches like DyanmiCoS, METEOR-S, and SeGSeC approaches. The use of semantic descriptions in order to automate composition in industry approaches like eFlow is still an awaiting result in web service composition field.

The dynamic web service composition problem is likely to be around with some open issues like supporting transactional support, verification of compositional correctness etc. In future research, addressing such issues will surely be beneficial and is desirable.

References

1. Srivastava, B., Koehler, J.: Web Service Composition - Current Solutions and Open Problems. In: ICAPS 2003 Workshop on Planning for Web Services. (2003) 28–35
2. Alonso, G.: Web services: concepts, architectures and applications. Springer Verlag (2004)
3. Shen, L., Li, F., Ren, S., Mu, Y.: Dynamic composition of web service based on coordination model. Advances in Web and Network Technologies, and Information Management (2007) 317–327
4. Tosic, V., Mennie, D., Pagurek, B.: Dynamic Service Composition and Its Applicability to E-Business Software Systems–The ICARIS Experience. Advances in Business Solutions (2002) 93–104
5. Dustdar, S., Schreiner, W.: A survey on web services composition. International Journal of Web and Grid Services 1 (2005) 1–30
6. Rao, J., Su, X.: A survey of automated web service composition methods. Semantic Web Services and Web Process Composition (2005) 43–54
7. Agarwal, V., Chafle, G., Mittal, S., Srivastava, B.: Understanding approaches for web service composition and execution. In: Proceedings of the 1st Bangalore annual Compute conference, ACM (2008) 1
8. Milanovic, N., Malek, M.: Current solutions for web service composition. IEEE Internet Computing 8 (2004) 51–59
9. Papazoglou, M.: Web services: principles and technology. Addison-Wesley (2008)
10. Papazoglou, M.: Web services and business transactions. World Wide Web 6 (2003) 49–91
11. ter Beek, M., Bucchiarone, A., Gnesi, S.: A survey on service composition approaches: From industrial standards to formal methods. Technical report, Technical Report 2006-TR-15 (2006)
12. Chen, Y., Li, Z., Jin, Q., Wang, C.: Study on qos driven web services composition. (Frontiers of WWW Research and Development-APWeb 2006) 702–707
13. Ran, S.: A model for web services discovery with QoS. ACM SIGecom Exchanges 4 (2003) 10
14. Papazoglou, M., Traverso, P., Dustdar, S., Leymann, F.: Service-oriented computing: State of the art and research challenges. Computer-IEEE computer society- 40 (2007) 38
15. Sun, H., Wang, X., Zhou, B., Zou, P.: Research and implementation of dynamic web services composition. Advanced Parallel Processing Technologies (2003) 457–466
16. Benatallah, B., Sheng, Q., Dumas, M.: The self-serv environment for web services composition. IEEE Internet Computing 7 (2003) 40–48
17. Zeng, L., Benatallah, B., Dumas, M., Kalagnanam, J., Sheng, Q.: Quality driven web services composition. In: Proceedings of the 12th international conference on World Wide Web, ACM (2003) 421

18. Weigand, H., van den Heuvel, W.J., Hiel, M.: Rule-based service composition and service-oriented business rule management. In Vanthienen, J., Hoppenbrouwers, S., eds.: Proceedings of the International Workshop on Regulations Modeling and Deployment (ReMoD'08), ACM (2008) 1–12
19. Charfi, A., Mezini, M.: Hybrid web service composition: business processes meet business rules. In: Proceedings of the 2nd international conference on Service oriented computing, ACM (2004) 30–38
20. Casati, F., Ilnicki, S., Jin, L., Krishnamoorthy, V., Shan, M.: eFlow: a platform for developing and managing composite e-services. Technical Report HPL-2000-36, HPL (2000)
21. Casati, F., Ilnicki, S., Jin, L., Krishnamoorthy, V., Shan, M.: Adaptive and dynamic service composition in eFlow. In: Advanced Information Systems Engineering, Springer (2000) 13–31
22. Aggarwal, R., Verma, K., Miller, J., Milnor, J.: Dynamic Web Service Composition in METEOR-S. Technical report, LSDIS Lab, Univeristy of Georgia, Athens (2004)
23. Pires, P., Benevides, M., Mattoso, M.: Building reliable web services compositions. Web, Web-Services, and Database Systems (2002) 59–72
24. Pires, P.F.: WEBTRANSACT: A Framework for Specifying and coordinating reliable web services compositions. Technical Report ES-578/02, Federal University of Rio De Janerio (2002)
25. Lécué, F., Silva, E., Ferreira Pires, L.: A framework for dynamic web services composition. Emerging Web Services Technology II (2007) 59–75
26. Silva, E., Ferreira Pires, L., van Sinderen, M.J.: Supporting Dynamic Service Composition at Runtime based on End-user Requirements. In: Workshop at the International Conference on Service Oriented Computing (ICSOC) 2009, Stockhome, Sweden. (2009) 22–27
27. Fujii, K., Suda, T.: Semantics-based dynamic Web service composition. International Journal of Cooperative Information Systems 15 (2006) 293–324
28. Tiwana, A., Ramesh, B.: E-services: Problems, opportunities, and digital platforms. In: Proceedings of the 34th Annual Hawaii International Conference on System Sciences, 2001. (2001) 8
29. Casati, F., Ilnicki, S., Jin, L.J., Krishnamoorthy, V., Shan, M.C.: An Open, Flexible, and Configurable System for E-Service Composition. Technical Report HPL-2000-41, HPL (2000)
30. Li, G., Hai, H., Hu, Z.: A Flexible Framework for Semi-automatic Web Services Composition. In: IEEE Asia-Pacific Services Computing Conference, 2008. APSCC'08. (2008) 1258–1262
31. Sivashanmugam, K., Miller, J., Sheth, A., Verma, K.: Framework for semantic web process composition. International Journal of Electronic Commerce 9 (2005) 71–106
32. Fujii, K., Suda, T.: Dynamic service composition using semantic information. In: Proceedings of the 2nd international conference on Service oriented computing, ACM (2004) 39–48