Service Composition Management Using Risk Analysis and Tracking

Shang-Pin Ma and Ching-Lung Yeh

Department of Computer Science and Engineering, National Taiwan Ocean University, Keelung, Taiwan
{albert,19957009}@ntou.edu.tw

Abstract. How to effectively and efficiently monitor, manage, and adapt web services is becoming a significant issue to address. In this paper, we argue that only solving emerging service faults at deployment time or runtime is not enough; on the contrary, we believe that prediction of service faults is equivalently important. We propose a risk-driven service composition management process including four main phases: preparation, planning, monitoring and reaction, and analysis. By applying the proposed approach, risky component services can be removed earlier, and the fault source can be tracked and identified more easily when any failure occurs. We believe the proposed risk-driven approach can effectively and efficiently ensure the robustness of an SOA-based system.

Keywords: service management, risk management, service composition.

1 Introduction

Service-Oriented Architecture (SOA) has become an important trend in software engineering for developing loosely-coupled applications and integrating legacy and modern systems. Accordingly, how to effectively and efficiently monitor, manage, and adapt web services is also becoming a significant issue to address. Today, many mechanisms [1, 2, 4, 5, 10, 12] are available to perform service monitoring and management for improving various types of QoS (Quality of Service), such as availability and reliability.

In this paper, we argue that only solving emerging service faults at deployment time or runtime is not enough; on the contrary, we believe that prediction of service faults is equivalently important. Risk analysis is an approach which is commonly used in project management domain [9]. The potential problems which may hinder the development of project are called risk. If the problems occur, the project will to be mired in difficulties. Therefore, the risk should be reduced before the project executing. At present, the risk management techniques are used in a variety of domains, such as electricity [7] and wireless security [6]. The risk concept is also applied to enhance the service-oriented design process to select appropriate business partners [8]. In this study, we bring the risk notion into the web service management mechanism to foresee possible problems or weak points in an SOA-based system.

C. Liu et al. (Eds.): ICSOC 2012, LNCS 7636, pp. 533–540, 2012.

Our proposed risk-driven service composition management process includes four main phases: preparation, planning, monitoring and reaction, as well as analysis. In preparation phase, the risk probability and the risk impact of each component service in a composite service are calculated based on historical QoS data. If any component service is too risky, another interface-compatible service will take over the risky one. Besides, a service dependency graph is also produced in this phase for tracking the fault source at runtime. In planning phase, a monitoring plan is generated based on the service dependency graph and the risk analysis result. In the monitoring and reaction phase, the composite service is monitored according to the monitoring plan. If any service failure occurs or a component service becomes too risky (i.e. risk exposure is larger than a given threshold), a fault tracking path is built immediately and automatically based on the service dependency graph, and appropriate reaction actions, such as re-invocation or service substitution, are chosen and performed in the light of the monitoring plan. Finally, in the analysis phase, all service execution logs are cumulated and analyzed to update the historical QoS database for further processing. By applying the proposed approach, risky component services can be removed earlier, and the fault source can be tracked and identified more easily when any failure occurs. We believe the proposed risk-driven approach can effectively and efficiently ensure the robustness of an SOA-based system.

The structure of the paper is as follows. Section 2 describes the details of the proposed service composition management approach, including core concepts and the proposed management process. Final section concludes this study.

2 Approach Descriptions

This section describes the proposed approach in detail, including the core concepts and the process of risk-driven service composition management.

2.1 Core Concepts for the Proposed Approach

For establishing the proposed service composition management mechanism, we introduce the significant concepts first by utilizing the UML class diagram and representing each concept as a class (shown in Figure 1). Composite service is the aggregation of multiple component services which are described by service profiles. The service profile is to obtain and store significant attributes of a component services, including service name, service interface, service type, service input, service output, and service provider. A Service Dependency Graph (SDG) can be produced based on the flow of a composite service to ease service fault tracking. The component service risk is consisting of risk impact and risk probability, which are calculated according to the composite service flow structure and the historical QoS (Quality of Service) data. Component service risk can be mitigated by performing replication or substitution actions for component services. The composite service risk is estimated by cumulating values of all component service risks.

A service flow instance is instantiated when the service requester utilizes a composite service. When any service failure occurs during execution of the service flow instance, we can track the Fault Tracking Path (FTP), which is consisting of a service fault source, intermediate nodes, and a service failure occurrence point, to find the cause of faults and to perform appropriate reaction actions. FTP is automatically generated based on the built SDG. Four reaction strategies are included in this study: Substitution, Compensation, Re-invoke, and Re-start. Substitution is a strategy that tries to select another service with the same service interface as the faulty one. The corresponding service interface of a service component is recorded in the service profile. Compensation is used to restore correct object states which were correct, in case these were affected by a fault. Re-invoke is re-executing the same service invocation with exactly the same parameters and contracts. Re-start is stopping and starting the web service server so that service may become available.

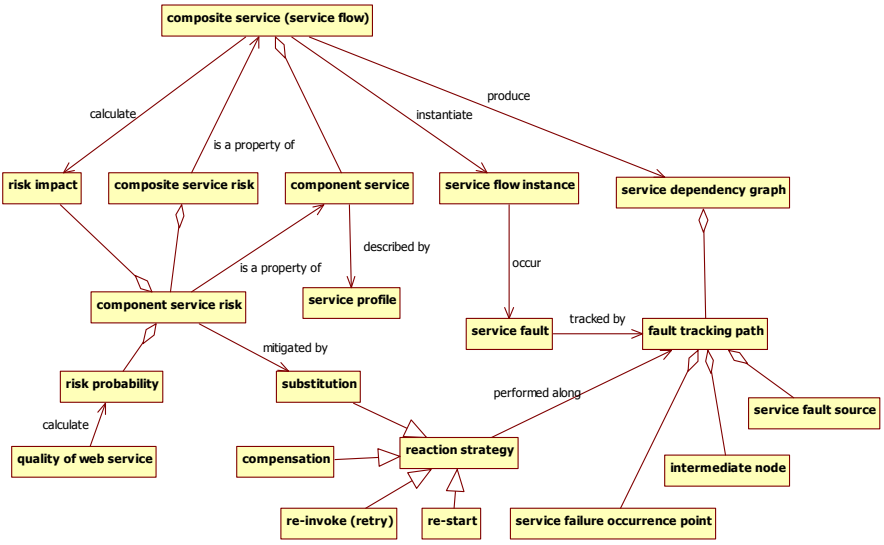

Fig. 1. Conceptual Model

Service Risk. Risk is the potential that may lead to damage. In our proposed approach, we bring the notion of risk to predict possibility faults or QoS decline by calculating risk exposure by equation (1).

$$Risk = Probability \times Impact \tag{1}$$

At preparation phase, difference mitigation actions, such as service replication or service substitution, will be executed according to the risk exposure level. At runtime, the risk exposure level is also an important indicator to carry out reaction strategies. Two elements of risk, risk probability and risk impact, are described as following.

Table 1. The Rating of Availability & Reliability

Rating	Detail
0.2	Both Availability and Reliability are more than 99.999%
0.4	Availability or Reliability is between 99.999% and 99.9%
0.6	Availability or Reliability is between 99.9% and 99%
0.8	Availability or Reliability is between 99% and 90%
1.0	Availability or Reliability is between 90% and 0%

Risk probability. Risk probability is a value to estimate the stability of a component service. We adopt the QoS value of the component service in historical data to calculate by equation (2):

$$P = \left[(1 - w_{\mathrm{p}}) \times T + w_p \times AR \right] \tag{2}$$

Where w_p is a weight value to represent the importance of AR value, AR is the rating of Availability & Reliability as showing in Table 1, and T is a value calculating by equation (3) for evaluating the stability of service response time.

$$T = \frac{Standard\ Deviation}{Arithmetic\ Mean} \tag{3}$$

Risk impact. Risk impact is a value to indicate the possible damage when a component service in composite service becomes malfunctional. In this study, risk impact is calculated by extending the method proposed in [11] since if a component service is more important, the damage is larger if this service cannot operate correctly. In this study, risk impact value is aggregating by the importance score assigned by the user and analysis results of execution path analysis by equation (4).

$$I = \frac{(1\text{-}w_i) \times I_e + w_{\mathrm{i}} \times I_u}{I_{\max}} \tag{4}$$

Where I_u is the score assigned by users (i.e. W_u value in [11]), I_e is the score calculated by execution path analysis (i.e. W_p value in [11]), w_i is the weight to represent importance of I_u in this equation, and I_{max} is the maximum of I values for all component services. Notably, through execution path analysis, we can assert that some of the component services play more important roles than others since these services emerged in more execution paths.

2.2 Risk-Driven Service Composition Management Process

Based on above concepts, we devise a management process, called Risk-Driven Service Composition Management (RDSCM), including four sub-processes: Preparation, Planning, Monitoring & Reaction, and Analysis. Differ from [2], we additionally define the preparation phase to prepare artifacts that can be leveraged when faults occur in following phases.

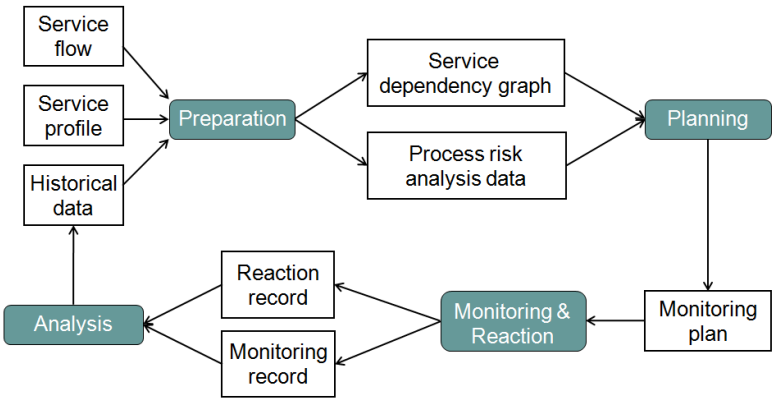

Fig. 2. Risk-Driven Service Composition Management Process

As shown in Figure 2, in preparation phase, RDSCM calculates the risk impact of each component service based on the service flow, and computes the risk probability of each component service according to historical QoS data. The risk exposure values can be determined by aggregating impact and probability data. Besides, RDSCM also analyzes the service flow and service profiles of all component services in the flow to generate the SDG (service dependency graph). In the planning phase, RDSCM produces a monitoring plan based on SDG and process risk analysis data. In the monitoring & reaction phase, RDSCM monitors the composite service according to the monitoring plan. If any fault occurs, RDSCM selects appropriate reaction actions specified in the monitoring plan. Finally, in the analysis phase, RDSCM analyzes all execution records and update the historical QoS data, which will influence risk analysis results of other instances of the same composite service or other composite services.

Preparation Process. As afore-mentioned, in the preparation phase, the risk of the component services which in the composite service are analyzed. If the risk of a component service is higher than a given threshold, RDSCM will try to select another service with the same service interface to replace the faulty one. The substitute strategy, such as selecting the service with best QoS or selecting the service used most frequently, can be decided in advance. If the risk of the substitute service is still too high, RDSCM will stop the process and show alert message.

In the preparation phase, SDG will be built. In opposite to the service flow showing the execution sequence, the SDG represents the dependency relationship among component services for a composite service. For constructing the dependency graph, we extract the data flow among component services, i.e. the source and the target of service data, and invert the service flow. For example, a process which has input and output data is shown in the left-hand side of Figure 3, and the dependency graph of this process is shown in the right-hand side. Due to the service S_2 accepts the input data o_1 from service S_1, we can assert S_2 depends on S_1, and the edge representing this dependency relationship in SDG is connected from source S_2 to target S_1. Following above graph construction rules, an SDG can be generated automatically.

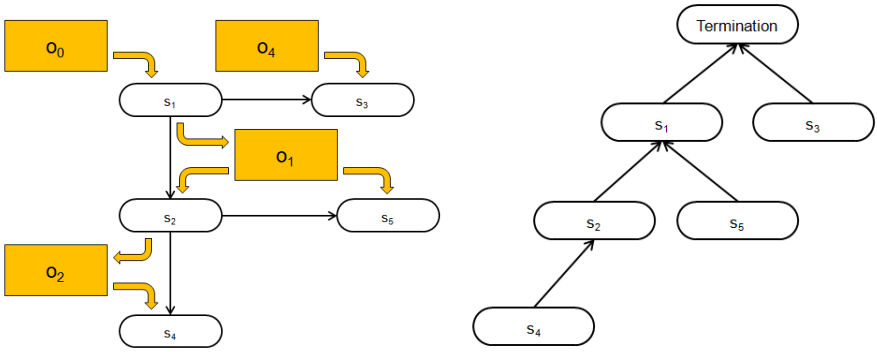

Fig. 2. Example: Building the Service Dependency Graph

Planning Process. In this phase, RDSCM automatically produces a monitoring plan in XML format, which can be modified by the manager. The plan includes four parts (1) Monitoring Attribute: to specify the QoS attributes which the manager plans to monitor; (2) Monitoring Threshold: to set the threshold of the high risk. (3) Monitoring Frequency: to arrange the monitoring frequency, i.e. the cycle time of QoS probing, for services with medium and low risk; and (4) Substitution Policy: to determine the procedure to swap the faulty or risky service. Available substitution policies include choosing the most used services, choosing the service with best QoS, and choosing the service with lowest risk.

Monitoring and Reaction Process. In this phase, RDSCM monitors the execution of the composite service based on the monitoring plan. This phase has two main tasks: first is collecting execution data with all service instances or additional service probing records according to the monitoring frequency setting; and second is detecting and recovering service faults.

If an instance reveals faults, RDSCM bases the SDG to produce a FTP (fault tracking path), and selects reaction strategies as well as performs recovery actions for component services which are in the fault tracking path. Following the error chain paradigm, a FTP includes the failure occurrence point, the fault source, zero or more intermediate services. Besides, zero or more non-failure services are preceding the fault source in the service flow. For example, as shown in Figure 3, if service S_4 is the failure occurrence point which reveals the fault, the possible longest FTP $\{S_4S_2S_1\}$ is established first due to any of these three services may be the fault source. If the service S_1 is confirmed as the fault source, the final FTP is also $\{S_4S_2S_1\}$ and service S_2 is assigned as an intermediate service since S_2 is either the failure occurrence point nor the fault source. If the service S_2 is determined as the fault source, the FTP is $\{S_2S_1\}$ without intermediate service.

When the FTP is produced, RDSCM can carry out the service recovery process to fix the process instance. RDSCM increases the risk of the fault source (and updates the value into database) first, and derives the service type of all services in the FTP for further processing. In this study, the service type can be identified from two

view-points: Retriable and Compensable (this notion is borrowed from [3]). If a component service guarantees a successfully termination after a finite number of invocations, it is Retriable (R); otherwise it is not retriable. If the component service which supports compensable transactions, it is Compensable (C); on the other hand, if this component service execution does not affect the state of the service, it is Non-compensatory (N). Depending on the type, RDSCM can fulfill appropriate reaction actions to recover the all services in the FTP. In the proposed recovery process, the fault source needs to recover first. For example, if the service type of the fault source is CR, RDSCM compensates this service at first, and then re-invokes it and tests if the service is workable. If the service is still malfunctioned, RDSCM selects another interface-compatible service according to the substitution policy to take over the faulty service. Next, RDSCM may compensate intermediate services or the failure occurrence point first (if the service is compensable) since the incorrect data may affect the business state of these services, and then re-invokes the service (if the service is retriable). Notably the reaction strategy of service substitution is unnecessary for non-fault-source services since these services are not malfunctional. Following above steps, all services in the FTP are recovered along the inverse sequence of the FTP and then the faulty composite service is recovered accordingly.

3 Conclusions

This paper presents a risk-driven approach to manage composite web services. The goal of this study is reduce the probability and damage of service faults occurring and further ensure the robustness of the SOA-based system, with the following key contributions:

- Bring the notion of risk management into web service management.
- Devise a method to calculate the service risk exposure for estimating the possible faults which reside in a service composition.
- Devise a method to construct the service dependency graph (SDG) and the fault tracking path (FTP) to efficiently track service faults and recover the faulty composite service.

Acknowledgements. This research was sponsored by National Science Council in Taiwan under the grant NSC 100-2221-E-019-037.

References

[1] Baresi, L., Guinea, S., Nano, O., Spanoudakis, G.: Comprehensive Monitoring of BPEL Processes. IEEE Internet Computing 14(3), 50–57 (2010)

[2] Calinescu, R., Grunske, L., Kwiatkowska, M., Mirandola, R., Tamburrelli, G.: Dynamic QoS Management and Optimization in Service-Based Systems. IEEE Transactions on Software Engineering 37(3), 387–409 (2011)

[3] El Haddad, J., Manouvrier, M., Ramirez, G., Rukoz, M.: QoS-Driven Selection of Web Services for Transactional Composition. In: 2008 IEEE International Conference on Web Services, ICWS 2008 (2008)
[4] Erradi, A., Maheshwari, P., Tosic, V.: WS-Policy based Monitoring of Composite Web Services. In: The Fifth European Conference on Web Services, pp. 99–108. IEEE Computer Society (2007)
[5] Friedrich, G., Fugini, M., Mussi, E., Pernici, B., Tagni, G.: Exception Handling for Repair in Service-Based Processes. IEEE Transactions on Software Engineering 36(2), 198–215 (2010)
[6] Hsin-Yi, T., Yu-Lun, H.: An Analytic Hierarchy Process-Based Risk Assessment Method for Wireless Networks. IEEE Transactions on Reliability 60(4), 801–816 (2011)
[7] Kettunen, J., Salo, A., Bunn, D.W.: Optimization of Electricity Retailer's Contract Portfolio Subject to Risk Preferences. IEEE Transactions on Power Systems 25(1), 117–128 (2010)
[8] Kokash, N.: Risk Management for Service-Oriented Systems. In: Baresi, L., Fraternali, P., Houben, G.-J. (eds.) ICWE 2007. LNCS, vol. 4607, pp. 563–568. Springer, Heidelberg (2007)
[9] Kwan, T.W., Leung, H.K.N.: A Risk Management Methodology for Project Risk Dependencies. IEEE Transactions on Software Engineering 37(5), 635–648 (2011)
[10] Lee, J., Ma, S.-P., Lee, S.-J., Wu, C.-L., Lee, C.-H.L.: Towards a High-Availability-Driven Service Composition Framework. In: Service Life Cycle Tools and Technologies: Methods, Trends and Advances, pp. 221–243. IGI Global (2012)
[11] Ma, S.-P., Kuo, J.-Y., Fanjiang, Y.-Y., Tung, C.-P., Huang, C.-Y.: Optimal service selection for composition based on weighted service flow and Genetic Algorithm. In: 2010 International Conference on Machine Learning and Cybernetics, ICMLC (2012)
[12] Moser, O., Rosenberg, F., Dustdar, S.: Non-intrusive monitoring and service adaptation for WS-BPEL. In: The 17th International Conference on World Wide Web, pp. 815–824. ACM, Beijing (2008)