Service and Protection Level Agreements for Business Processes *

Ganna Frankova and Artsiom Yautsiukhin

Dept. of Information and Communication Technology - University of Trento
email: {frankova,evtiukhi}@dit.unitn.it

Abstract. The issue of business process design for a complex web service provision is gaining attention and has been addressed in a number of recent works. We argue that calculation of global quality of service and protection, which then is negotiated with a client as service and protection level agreements, for complex web services must be based on business process.
The quality of service in web service compositions plays a vital role and has opened a wide spectrum of challenges. Therefore, the orchestrator should design its business process aggregating web services in such a way as to make it more efficient from the quality of service and protection point of view. In this work we propose a methodology that identifies the concrete business process providing the highest quality of service and protection among all possible design alternatives.

1 Introduction

One of the most thought provoking issues in web services is that of building a business process to provide a complex added-value service. A Business Process (BP) in a web services context is a set of interrelated services and the flow of data between them that leads to the outcomes associated with a business activity. The research on business processes is well under way. There are many works focusing both on functional properties, primary goals, e.g., [7], and non-functional properties, quality of service, of web services and web services composition, e.g., [10]. However, the works do not take into account the concept of business process for web service composition. The works on security for web services are mainly restricted to protection of communication links [8] and access control [2]. There are only a few approaches covering security requirements for web services as a whole [3, 6].

The Quality of Service (QoS), e.g., execution time and availability, and Quality of Protection (QoP), e.g., time to recover after an attack and the percentage of successful virus attacks are of paramount importance for the success of web services. Web services should satisfy not only functional requirements but also be QoS/QoP driven. The problem becomes even more complex if we deal with composite web services.

To design a BP first we specify an abstract BP which fulfils the desired functional requirements at the high level. Various design alternatives that do not

* This work has been partly supported by the IST-FP6-IP-SERENITY and IST-FP6-IP-SENSORIA projects.

affect the functionality of abstract business process but allow configuring the workflow in different ways exist (e.g., one alternative provides faster processing, another one is more reliable). These design alternatives lead to several concrete BPs (refinements of the abstract BP) that precisely define all its steps at the low level and provide all the functionalities. Then, the most efficient concrete BP from the QoS and QoP point of view is selected. Furthermore, an orchestrator may trust one service provider more than another one on provision of a service that is a part of a concrete BP.

In this work, we propose a methodology that identifies the concrete business process providing the most fruitful qualities among all possible design alternatives for the given abstract business process. Moreover, the methodology takes into account the level of trust of service providers and adjusts the expected quality value correspondingly. The methodology is supported by a reasoning algorithm that allows easy recalculation of computed metrics if some changes in BP design occur, that is typical for highly dynamic environment such as web services.

2 Service and Protection Level Agreements: Background

The stake-holders involved in a BP are:

Definition 1 *Service Customer* (customer) is an entity that interacts with a complete, self-contained BP. *Service Orchestrator* (orchestrator) is an entity which manages a BP and agrees to satisfy the customer's requirements for the BP. *Service Provider* (provider) is an entity that has a task assignment, i.e., web service, that is a part of a higher-level BP, received from an orchestrator.

The involved partners should come to a formal agreement before the usage of web service. The agreement is defined as a contract between the provider and the orchestrator specifying the functionality of the outsourced service, quality and protection requirements. The requirements for complex web service the orchestrator manages are specified in a contract with a customer.

Here we assume that services provide desired functionality and we focus on non-functional requirements. We found it useful to divide the agreement into the following parts:

Definition 2 *Service Level Agreement* (SLA) is a contractual version of the Quality of Service (QoS) which specifies the performance criteria a provider promises to meet while delivering a service. *Protection Level Agreement* (PLA) is a contractual version of the Quality of Protection (QoP) which specifies security criteria, a provider promises to meet while delivering a service.

We argue that security requirements should be considered since client's data may be corrupted while under the control of the provider. On the one hand, there are plenty of works on QoS [9, 5], on the other hand, very little attention is devoted to QoP while identification and aggregation of security metrics useful for QoP is not a trivial task [6].

3 Business process hypergraph

The cornerstone of web services success lies in the ability to compose web services in order to build complex added-value services. Dealing with quality aware web service composition requires studying and finding the global SLA/PLA of complex web services according to the BP. In the proposed methodology, we use hypergraphs [1] to capture the structure of BP. We introduce the notion of business process hypergraphs as follows.

Definition 3 A **business process hypergraph** (BPH) $\mathcal{B}$ is a pair $\langle S, D\rangle$ where S is a set of service requirements and D is a set of hyperarcs. A *hyperarc* is an ordered pair $\langle N, t\rangle$ from an arbitrary nonempty set $N \subseteq S$ (source set) to a single node $t \in N$ (target node). Each hyperarc is associated with a weight $\omega_{\langle N,t\rangle}$ and a function $\varphi_{\langle N,t\rangle}$ which calculates value of a target node taking as arguments source nodes and the weight $\omega_{\langle N,t\rangle}$.

Since in our case each source node contributes differently to a target one, we use dummy nodes between source and target nodes, one for a hyperarc. The weights are assigned to the hyperarcs that connect source and dummy nodes and the weights for the hyperarcs between the dummy and target nodes are always 1. We do not depict the nodes to avoid unnecessary complexity.

We assume that there are more then one ways to implement an abstract BP. Each solution is given by the hyperpath defined as follows.

Definition 4 Let $\mathcal{B} = \langle S, D\rangle$ be a BPH, $X \subseteq S$ be a non-empty subset of services, and y be a service in S. A **hyperpath** $\mathcal{D}_{X,y}$ from X to y in $\mathcal{B}$ is a set of hyperarcs such that either $y \subseteq X$ or there exists a hyperarc $\langle Z, y\rangle \in D$ and there are hyperpaths from X to each service $z_i \in Z$.

As there are several ways to refine an abstract BP into the concrete BP, several hyperpaths exist. The global SLA/PLA of each hyperpath is different. Hence, the key issue is to determine the "minimal" hyperpath through a quantitative evaluation of BPH.

4 SLA and PLA for Business Processes: Methodology

The proposed methodology helps a service orchestrator to select the optimal concrete BP with the preferred QoS/QoP from several alternatives based on the abstract BP [1].

We made the following assumptions about a BP our methodology deals with: (i) BP is defined in a hierarchical way. A top level BP (BP_t) is based on services aggregated by one of structural activities such as a sequence, a loop, a choice or a parallel execution. Then, for each non-atomic service S_i a BP (BP_{S_i}) is determined and the decomposition continues until atomic, i.e, non-decomposing, services are reached. (ii) An orchestrator which does not trust a provider on

[1] For the notions of abstract business processes we refer to Web Services Business Process Execution Language Version 2.0, August 2006.

achievement of some requirements may use the level of trust to adjust corresponding metrics in such a way that after the modification the trust relation is established.

The methodology includes three phases, namely, (1) business process hypergraph construction, (2) aggregation functions design and (3) reasoning in business process hypergraph.

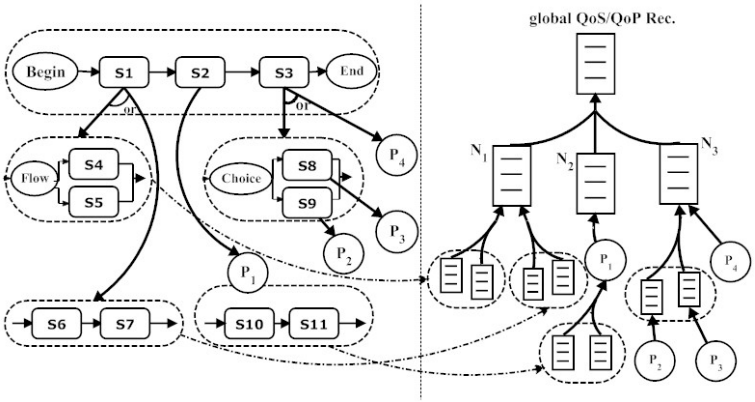

Fig. 1. Business process hypergraph construction.

Phase 1. Business Process Hypergraph Construction. The first phase of the methodology is devoted to BPH construction based on the various implementation of abstract BP designing by an orchestrator. The BPH construction process is presented as an algorithm in Figure 2.

Figure 1 shows the top level BP_t based on three services S_1, S_2, and S_3. Each service of the top level BP_t is associated with the node N_1, N_2, and N_3. The nodes are connected by one hyperarc with the top node, that means that the services all together contribute to satisfaction of the global requirements. Then, the services are decomposed. Since two alternative BPs exist for service S_1, two distinct hyperarcs and corresponding source nodes are added in the hypergraph. This means that each alternative business process contributes to the satisfaction of the target requirements separately. The services are executed by the same partner that runs BP_t. Service S_2 is delegated to provider P_1 (shown as a node in the BPH) and then decomposed. Service S_3 may be fulfilled by provider P_4 or may be decomposed into a business process whose activities are outsourced to P_2 and P_3.

Phase 2. Aggregation Functions Design. The second phase of the methodology is devoted to the aggregation functions. Each hyperarc is assigned with a set of weights that shows contribution of a source node to the target one. Weights of hyperarcs connecting providers with a target node denote the level of trust between the delegator and the delegatee. Each leaf node is assigned with a QoS/QoP value that *can* be achieved. The value corresponds to the QoS/QoP of atomic service.

Each hyperarc is assigned with an aggregation function $\varphi_{\langle N,t\rangle}$ (one for each structural activity) which calculates the value of a target node taking as argu-

ments the source nodes and the set of weights $\omega_{\langle N,t \rangle}$. Examples of the aggregation functions for QoS could be found in [5]. The authors provide aggregation functions for such numerical QoS metrics as cost, execution time, etc. In our work we take into account the level of trust of providers and propose an additional function to change the expected metrics according to the level of trust.

```
Algorithm Business Process Hypergraph Construction
begin
   specify the global QoS/QoP requirements for BP
   associate the global QoS/QoP requirements for BP with the top node of BPH
   while there are services with the corresponding decomposing BPs
      if service is outsourced then
         for each provider that has a task assignment
            add a node for provider in the BPH
            add a hyperarc from the added node to the target node
      else
         for each design alternative
            for each service of the alternative
               add a node in the BPH
            add a hyperarc from the added nodes to the target node
   end-while
end
```

Fig. 2. Business Process Hypergraph Construction

To the best of our knowledge there is no similar approach for security metrics and we propose the preliminary aggregation functions for number of attacks per month in the following table.

Structural activities	**Weight**	**Function**
parallel	probability of service execution[2]	$\varphi = \sum_{x_i} \omega_i * x_i$
sequence	probability of service execution[2]	$\varphi = \sum_{x_i} \omega_i * x_i$
choice	probability of service execution[2]	$\varphi = \sum_{x_i} \omega_i * x_i$
loop	$\omega = 1$	$\varphi = \varphi_1$
trust	level of trust	$\varphi = \omega_1 * \varphi_1$

Phase 3. Reasoning in Business Process Hypergraph. The third phase of the methodology is devoted to a reasoning algorithm that proceeds several concrete BPs calculating their QoS/QoP. We evaluate each requirement separately finding the "best" BP for each requirement rather than for all of them together.

The aggregation functions proposed by Jaeger [5] are superior. This allows us to apply one of already proposed algorithms for effective calculation of hyperpath in BPH [1, 4]. Quantitative evaluation of hyperpaths in BPH from leaf nodes to the top one determines the optimal concrete BP.

We note that QoP metrics are more complex than QoS ones and aggregation functions for QoP metrics design is not a trivial task. We developed and proved

[2] probability to find a source service to be executed if the target service is proceeding.

an algorithm for calculation of an optimal hyperpath that works with monotone aggregation functions. Due to lack of space we do not present it in this work.

5 Concluding Remarks

We have proposed a methodology that during design time helps a service orchestrator to determine the optimal concrete BP from several alternatives according to the preferred QoS/QoP. The methodology also allows the orchestrator to easily recalculate computed metrics if some changes in the BP occur (i.e., BP re-design). Furthermore, we are planning to adopt the approach for automatic composition of BP (i.e. automatic BP design) to support a business process planning tool in choosing the best concrete business process on the fly.

In the future, we will investigate the multi-requirement analysis, which requires the identification of a decision-making function that chooses the more preferable *set* of attributes (e.g. a weighted function). We also will define and validate the aggregation functions for more QoS/QoP requirements. Furthermore, to make the discussion more concrete, we will justify that the aggregation functions are appropriate using the e-business banking case study, a working scenario of the IST-FP6-IP-SERENITY project. In addition, we plan to elaborate the issue of trust to determine how it affects the expected qualities.

6 Acknowledgments

We thank Fabio Massacci for support in conducting this research. We acknowledge Marco Aiello for his constructive suggestions for improving the quality of the paper.

References

1. G. Ausiello, G. F. Italiano, and U. Nanni. Optimal Traversal of Directed Hypergraphs. Technical Report TR-92-073, International Computer Science Institute, Berkeley, CA, 1992.
2. E. Bertino, J. Crampton, and F. Paci. Access Control and Authorization Constraints for WS-BPEL. In *Proceedings of IEEE ICWS.*, 2006.
3. V. Casola, A. Mazzeo, N. Mazzocca, and M. Rak. A SLA Evaluation Methodology in Service Oriented Architectures. In *Proceedings of QoP Workshop.*, 2005.
4. G. Gallo, G. Longo, S. Pallottino, and S. Nguyen. Directed Hypergraphs and Applications. *Discrete Applied Mathematics*, 42(2-3):177–201, 1993.
5. M.C. Jaeger, G. Rojec-Goldmann and G. Mühl. QoS Aggregation in Web Service Compositions. In *Proceedings of IEEE EEE.*, 2005.
6. Y. Karabulut, F. Kerschbaum, P. Robinson, F. Massacci, and A. Yautsiukhin. Security and Trust in IT Business Outsourcing: a Manifesto. In *Proceedings of STM Workshop*, 2006.
7. N. Milanovic and M. Malek Current Solutions for Web Service Composition. *IEEE Internet Computing*, 8(6):51–59, November/December 2004.
8. OASIS Web Services Security: SOAP Message Security 1.1, February 2006.
9. T. Yu and K.-J. Lin. A Broker-Based Framework for QoS-Aware Web Service Composition. In *Proceedings of the IEEE EEE*, 2005.
10. L. Zeng, B. Benatallah, M. Dumas, J. Kalagnanam, and Q.Z. Sheng. Quality Driven Web Services Composition. In *Proceedings of WWW*, 2003.