Towards Choreography Transactions

Oliver Kopp, Matthias Wieland, and Frank Leymann

Institute of Architecture of Application Systems, University of Stuttgart, Germany
Universitätsstraße 38, 70569 Stuttgart, Germany
{lastname}@iaas.uni-stuttgart.de

Abstract. The focus of choreography modeling is to capture the message exchange between processes. Common choreography modeling languages do not provide capabilities to group activities of different participants together into an all-or-nothing group. This paper presents choreography spheres as a modeling technique for cross-process transactions based on BPEL4Chor and sketches a mapping to BPEL.

1 Introduction

BPMN [1] is an established standard for modeling processes and process choreographies. While BPMN is mostly used for modeling business processes, BPEL [2] is used for executing processes. BPEL itself defines orchestration only. Support for process choreographies is added by BPEL4Chor [3]. Since the execution semantics of BPMN is still not defined for all cases, we focus on BPEL4Chor, which has an agreed semantics [4].

In BPEL4Chor a choreography is modeled by modeling the behavior of each participant as BPEL process and interconnecting the pools using message links. Internal transaction behavior is modeled in BPEL by scopes. A scope groups activities together. If a scope is completed, its whole work may be compensated as a response to failures in subsequent steps. The compensation is either performed by explicitly modeled compensation behavior or by the default compensation. The default compensation executes the (explicitly modeled) compensation actions for each activity in reverse order of their execution. Thus, an all-or-nothing behavior may be achieved even for activities not providing an "undo" operation themselves. However, there exist scenarios where it is helpful for process modelers to use modeling constructs spanning over different processes to express that the selected activities of different processes belong together and have to be coordinated. Usually, the coordination has to be modeled manually including the activities needed to communicate with a coordinator. The contribution of this paper is the introduction of *choreography spheres*, which represent this missing construct. The main advantage of using choreography spheres is that the coordination between the participant does not have to be modeled explicitly. By using choreography spheres, the coordination is done automatically. We present two distinct types of choreography spheres, each being a new modeling construct for the two common transaction types: short running and long running transactions.

In general, we define a choreography sphere as shown in Definition 1.

Definition 1 (Choreography Sphere). *A choreography sphere ensures transactional behavior of all enclosed activities that can be part of multiple processes.*

The definition leads to a all or nothing semantic of the sphere, which means all activities are either executed successfully or the effects are undone.

In the following, an overview on background and related work is given in Section 2. Afterwards, two types of spheres with different transaction properties are described in Section 3 and Section 4. Finally, Section 5 concludes and presents an outlook on future work.

2 Background and Related Work

The concept of spheres with transactional behavior was first introduced in [5]. The spheres were allowed to overlap, but the semantics for choreography spheres was put as future work. Currently, there exist no work on cross-organizational transactions, where arbitrary activities may be chosen to be coordinated.

There are two different types of transaction styles available in the the business area: business transactions and ACID style transactions [6]. Both are supported by spheres. But they have different implications on the modeling and also the technical coordination needed. Because of that we define in this paper 2PC Spheres in Section 3 for the ACID transactions and the BPEL Sphere in Section 4 for the Business transactions.

In the field of Web services, the standards published by the OASIS Web Services Transaction (WS-TX) Technical Committee are the standards used in practice. They provide transaction support across different vendors. The WS-Coordination specification [7] describes the general transaction framework. It describes the protocols for participant registration and defines a transaction context. This context can be passed using messages to enable the recipient to register as participant of the same transaction. WS-Coordination describes a protocol service, which handles the concrete coordination protocols. WS-Coordination is open to any transaction protocol. Up to now, WS-Atomic Transaction (WS-AT, [8]) and WS-Business Activity (WS-BA, [9]) are defined. WS-AT defines the two-phase-commit protocol, which is used for ACID transactions and thus provides and all-or-nothing behavior [10]. WS-BA is used for long-running transactions, which may last for years. It builds on the Saga model [11], where each activity has an associated compensation activity. In Saga, the activities are executed one after another. If an activity fails, the executed activities are compensated in reverse order. An overview of the history of transaction handling and current approaches in the field of Web services is given in [12].

It is shown in [13] how transactional behavior of services called by a BPEL process can be enforced. A transaction policy is attached to a group of BPEL activities. The transaction policy states which transaction protocol is used at invoking the grouped activities. We re-use this idea in our approach to enable transaction context propagation from the choreography sphere to the called services.

A variant of cross-process spheres is presented in [14], called "split BPEL scopes". These scopes result from a split of a BPEL process among different participants. The split BPEL scopes keep the semantics of the BPEL scope in the unsplit process and are coordinated using the WS-Coordination infrastructure [15]. Our approach allows for picking arbitrary activities in a choreography to be coordinated.

In the field of WS-Coordination, transaction protocols are defined using a coordination protocol graph and additional textual description. The approach presented in [16] shows how the coordination protocol graphs can be transformed to an abstract BPEL process for the coordinator and one BPEL process for the participant. Each of them has to be manually refined to adhere the requirements of the textual description of the transaction protocol. This has to be done only once for each transaction protocol. We use the resulting executable BPEL coordination logic in our approach as described in Section 3.

3 2PC Spheres

The two-phase commit protocol (2PC for short) is a well-known protocol to coordinate distributed ACID transactions enabling an all-or-nothing behavior. Due to their nature, ACID transactions are used for short-term operations (a few seconds). The two booking workflows presented on the left side (participant behavior descriptions) in Fig. 1 are executed in parallel, but need to be coordinated to ensure an all-or-nothing semantics of the two pay activities. To achieve that they are nested in a 2PC sphere, which ensures that no money is transferred if one activity fails. Only if both are executed successfully the whole sphere is

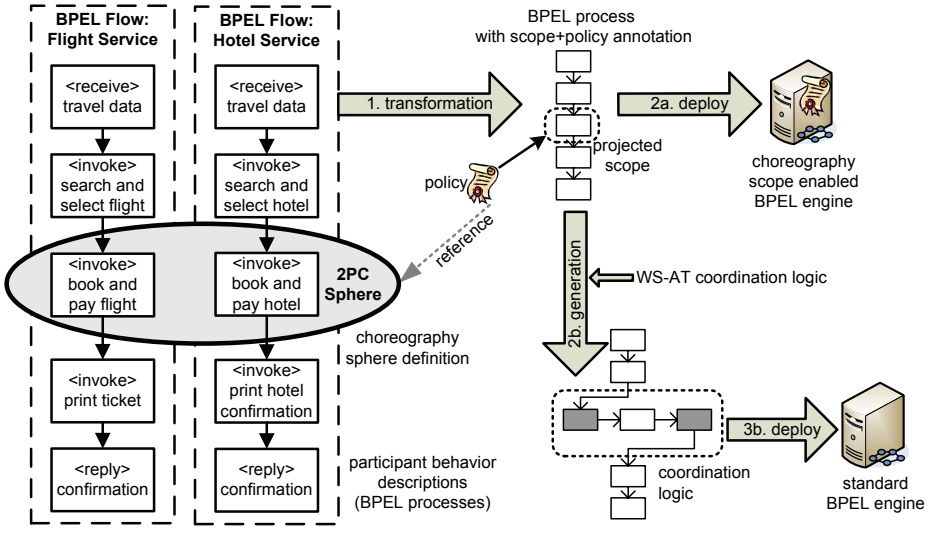

Fig. 1. Transformation steps required for the execution of choreography spheres

committed. Thus, one is sure that no money needs to be claimed back in any case of failure. Until now, 2PC spheres are defined for the flow activity only and cannot be nested.

The sphere definition has to be represented in each BPEL process to enable proper execution. The necessary steps for that are presented in Fig. 1. In step 1, each participant behavior description is transformed to a BPEL process, where a scope is put around the activities belonging to the choreography scope. The scope is annotated with a policy stating that the scope belongs to a choreography scope. In step 2a, each process is deployed on a choreography scope enabled BPEL engine. Step 2b is motivated by the non-existence of BPEL engines supporting choreography scopes. Thus, the alternative approach is to generate the coordination logic into each BPEL process. Finally, step 3 deploys each executable process on a standard BPEL engine.

4 BPEL Spheres

In contrast to short-running transactions, business transactions are usually long-running [17] as shown in example presented in Fig. 2. In this example a production company plans a new product and orders the parts for that product from different subcontractors. They produce the parts and send them to the production company that assembles them to the end product. In case of a fault in the inner BPEL sphere, the production company tires to find a new subcontractor. If this is not possible or any other error is propagated to the outer BPEL sphere, all running activities inside the outer BPEL sphere are terminated and all successful completed activities are compensated. The inner BPEL sphere is treated as child

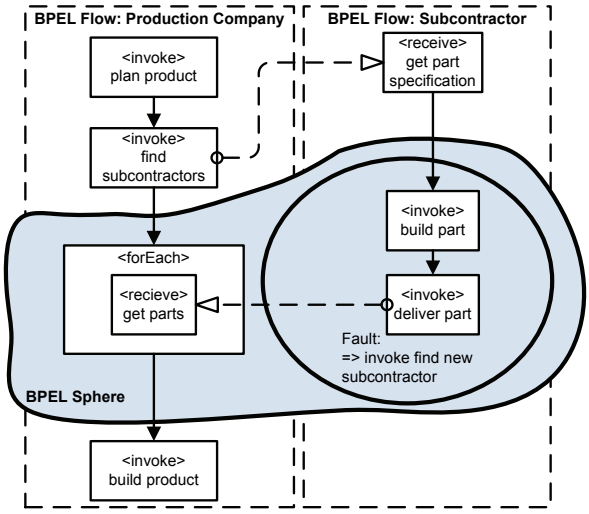

Fig. 2. BPEL sphere extending BPEL's scope semantics across BPEL processes

activity of the outer sphere and is handled the same way. This kind of process is running significantly longer than the money transfer actions presented in Section 3. Due to the long processing time, it is not possible to lock all data used in the processes. Thus, WS-AT cannot be used for the involved Web services as done in the case of 2PS spheres. The transformation steps presented in Section 3 are similar for BPEL spheres. The only change is that the used coordination protocol changes from WS-AT to WS-BA.

In BPEL, long-running transactions are realized by scopes. The concept of BPEL spheres extends BPEL's scope semantics to choreographies. A BPEL sphere groups activities together to form a transactional unit. The BPEL sphere is a long-running sphere and therefore uses compensation as undo operation.

Each BPEL sphere may have fault handlers and a compensation handler attached. BPEL spheres must be properly nested similar to scopes. BPEL spheres may not overlap with BPEL scopes similar to 2PC spheres. The activities in these handlers must be annotated which the participant, where the respective activities run. After the choreography is defined and it comes to the mapping to separate BPEL processes, the activities inside the handlers are split as presented in [15]. The remainder of the transformation and the deployment follows the procedure presented in Fig. 1. The projected scopes are annotated with a policy including the reference to the choreography scope definition.

It is possible to require that at least one activity in each participant runs in a BPEL sphere. This requirement ensures that all projected scopes run, which in turn is a requirement resulting from the premises by [14]. Thus, the projected BPEL scopes can be coordinated using the protocols described in [14]. An implementation of BPEL spheres has been described in [18], which is based on the pluggable framework [19].

5 Conclusion and Outlook

In this paper, we presented the concept of choreography spheres, where arbitrary activities of different processes can be grouped together. We showed a sketch of a possible implementation using the BPEL and WS-Coordination. We plan to add a full runtime support for choreography spheres to the Apache ODE engine.

Currently, choreography spheres may not overlap and the included activities may only be part of the same scope and loop. We sketched the nesting of BPEL spheres and the interplay with local scopes. A semantics for overlapping spheres is shown in [5]. Thus, our future work is to study possible semantics of overlapping BPEL spheres and of BPEL spheres overlapping with local BPEL scopes.

Motivated by [6,20], we research whether there are requirements for additional types of spheres. We are going to evaluate the applicability of these types in choreography settings.

Acknowledgments This work is supported by the BMBF funded project Tools4BPEL (01ISE08B) and the DFG project Nexus (SFB627).

References

1. Object Management Group (OMG): Business Process Modeling Notation (BPMN) Version 1.2. (2009) `http://www.bpmn.org/`.
2. OASIS: Web Services Business Process Execution Language Version 2.0 – OASIS Standard. (2007)
3. Decker, G., Kopp, O., Leymann, F., Weske, M.: BPEL4Chor: Extending BPEL for Modeling Choreographies. In: ICWS. (2007)
4. Lohmann, N., Kopp, O., Leymann, F., Reisig, W.: Analyzing BPEL4Chor: Verification and Participant Synthesis. In: WS-FM. (2007)
5. Leymann, F.: Supporting Business Transactions Via Partial Backward Recovery In Workflow Management Systems. In Lausen, G., ed.: BTW. (1995)
6. Greenfield, P., Fekete, A., Jang, J., Kuo, D.: Compensation is Not Enough. In: EDOC, Washington, DC, USA (2003)
7. OASIS: Web Services Coordination (WS-Coordination) Version 1.1. (2007)
8. OASIS: Web Services Atomic Transaction (WS-AtomicTransaction) Version 1.1. (2007)
9. OASIS: Web Services Business Activity (WS-BusinessActivity) Version 1.1. (2007)
10. Gray, J., Reuter, A.: Transaction Processing: Concepts and Techniques. Morgan Kaufman (1993)
11. Garcia-Molina, H., Salem, K.: Sagas. In: SIGMOD. (1987)
12. Wang, T., Vonk, J., Kratz, B., Grefen, P.: A survey on the history of transaction management: from flat to grid transactions. Distributed and Parallel Databases **23**(3) (2008) 235–270
13. Tai, S., Khalaf, R., Mikalsen, T.A.: Composition of Coordinated Web Services. In Jacobsen, H.A., ed.: Middleware. (2004)
14. Khalaf, R., Leymann, F.: Coordination Protocols for Split BPEL Loops and Scopes. Technical Report Computer Science 2007/01, University of Stuttgart, Institute of Architecture of Application Systems (2007)
15. Khalaf, R.: Supporting business process fragmentation while maintaining operational semantics: a BPEL perspective. Doctoral thesis, University of Stuttgart, Faculty of Computer Science, Electrical Engineering, and Information Technology, Germany (2008)
16. Kopp, O., Wetzstein, B., Mietzner, R., Pottinger, S., Karastoyanova, D., Leymann, F.: A Model-Driven Approach to Implementing Coordination Protocols in BPEL. In: MDE4BPM. (2008)
17. Leymann, F., Roller, D.: Production Workflow – Concepts and Techniques. Prentice Hall PTR (2000)
18. Steinmetz, T.: Ein Event-Modell für WS-BPEL 2.0 und dessen Realisierung in Apache ODE. Diploma thesis, University of Stuttgart, Faculty of Computer Science, Electrical Engineering, and Information Technology, Germany (2008) *(In German).*
19. Khalaf, R., Karastoyanova, D., Leymann, F.: Pluggable Framework for Enabling the Execution of Extended BPEL Behavior. In: WESOA. (2007)
20. Leymann, F., Pottinger, S.: Rethinking the Coordination Models of WS-Coordination and WS-CF. In: ECOWS. (2005)