The Influence of an External Transaction on a BPEL Scope

Oliver Kopp, Ralph Mietzner, and Frank Leymann

Institute of Architecture of Application Systems, University of Stuttgart, Germany
lastname@iaas.uni-stuttgart.de

Abstract. Business processes constitute an integral part of today's IT applications. They contain transactions as essential building blocks to ensure integrity and all-or-nothing behavior. The Business Process Execution Language is the dominant standard for modeling and execution of business processes in a Web service environment. BPEL itself contains a transaction model based on compensation, that describes the (local) transactions in a business process. The WS-Coordination framework deals with (external) transactions between Web services and is used to define the transaction behavior between a BPEL process and its partners. In this paper, we investigate how external transactions between Web services interrelate with local transactions of BPEL.

1 Introduction

The Web Services Business Process Execution [1] language (BPEL for short) is the dominant language used for defining business processes based on Web services. BPEL is designed to support business processes that are long-running, possibly running for years. Since a human may use other programs than data bases, the traditional ACID principle cannot be used. Therefore, a compensation based principle has been introduced and is used in BPEL. Activities belonging together are grouped by a BPEL 'scope'. A scope offers the possibility to attach a compensation handler to the grouped activities. This compensation handler is run if the scope should be compensated.

Transactions internal to business processes are well understood. BPEL, however, is agnostic to transaction behavior offered by calling services: Current BPEL engines and workflow languages do not support a scope being compensated from outside. This paper presents the first concept, where a local transaction (marked as a scope) can be part of two transactions: the local transaction and an external transaction.

Consequently, the paper is structured as follows: Section 2 provides an overview of related work in the field. Section 3 deals with the case, where a scope is part of a partner's transactions and presents necessary extensions to current coordination protocols. Finally, Section 4 concludes and provides an outlook on future work.

R. Meersman, T. Dillon, P. Herrero (Eds.): OTM 2009, Part I, LNCS 5870, pp. 381–388, 2009.

2 Background and Related Work

A history of transactions in workflows is presented in [2,3]. The only work allowing overlapping spheres of transaction is presented in [4]. That work specifies semantics for completed spheres, but neither specifies consequences of a fault in a sphere nor specifies coordination protocols.

In general, there are two different types of transaction styles available in the business area: compensation-based transactions and ACID style transactions [3]. In the field of Web services, WS-Coordination [5] and its protocol specifications are the de jure standard for coordinating transactions [6]. WS-C offers WS-Atomic Transactions (WS-AT, [7]) for 2PC transactions and WS-Business Activity (WS-BA, [8]) for compensation based transactions. WS-Coordination itself specifies an extensible framework for service coordination. It offers three types of services: activation service, registration service and protocol service. The initiator of a coordination requests a coordination context from the activation service. This context is used to distinguish the concrete coordination from other coordinations running in parallel and contains an endpoint reference for the registration service. The context is sent to the participants, which register themselves at the registration service, which returns an endpoint for a protocol service used for the concrete coordination. WS-Coordination is not restricted to a fixed set of possible coordination protocols, since it allows the use of arbitrary coordination protocols. This extensibility has been used to offer coordination for split BPEL process [9], coordinating auctions [10] as well as externalizing the coordination of BPEL scopes as a whole [11].

It is shown in [12] how WS-Coordination (WS-C, [5]) can be used to include services in local transactions. In the case of a BPEL process being completely under control of a caller, [13] shows the necessary coordination protocols.

A comparison of WS-BA and the transaction model of BPEL is provided in [14]. In summary, the transaction model of BPEL and WS-BA follow the Saga principle, but they differ in the treatment of external parties: BPEL (without the extensions presented in [11,12]) only sees sub-scopes as participants of a transaction and does not forward the transaction context to other services nor allows for inclusion of a scope into an external transaction.

A general classification of workflow and transaction languages is presented in [15]. BPEL and the related WS-Coordination specifications belong into the class *Tx+WF*, where "workflow and transaction models exist at the same level" [15].

[16] shows how a coordinator can use completion rules to automatically decide whether a WS-BA transaction should be completed or compensated. This work can be integrated in our work to gain a more sophisticated coordinator for deciding on the completion of the transaction.

Criticism on BPEL's transaction model is presented in [17,18]. The main point of criticism is that a compensation handler may only be called at a fault handler. Even if this issue was solved, the work did not present solutions for the interplay of local and external transactions.

Besides WS-Coordination, the business transaction protocol (BTP, [19]) and the Web Services Composite Application Framework (WS-CAF, [20]) are relevant. While BTP extends the 2PC protocol towards long-running transactions, WS-CAF offers a true compensation-based protocol. A comparison between WS-Coordination, BTP and WS-CAF is presented in [20]. It is shown in [21] that BTP and WS-CAF can be mapped to WS-Coordination. In the context of BPEL, WS-Coordination is the de facto standard and we show how WS-Coordination can be used to coordinate internal and external transactions.

UN/CEFACT's Modeling Methodology (UMM, [22]) offers a way to model business transactions. A business transaction is a collaboration between two parties and does not include compensation and ACID transaction concepts. It is shown in [23] how UMM Business Transactions can be transformed into abstract BPEL processes. Faults of BPEL scopes are communicated to the partner using explicit messages and not by an underlying transaction protocol.

Interactions between processes are captured in process choreographies [24]. A choreography description consists of a list of all participants, the messages exchanged and the ordering of the message exchange. There are several choreography modeling languages available [25]. The most prominent are WS-CDL [26] and BPMN [27]. WS-CDL mentions "transaction protocols" but does not state how these can be applied in the choreography. BPMN only regards local transactions and does not regard cross-process transactions. Currently, BPEL4Chor [25] is the only language extending BPEL towards choreography modeling. Cross-process transactions are added to BPEL4Chor in [28] by introducing the concept of a choreography sphere, where activities of arbitrary participants are marked to belong to one transaction. The work regards choreography spheres as the only inter-process transaction dependency and does not regard influences of a scope to activities nested in a partner's scope.

3 Process Partially Controlled by an External Partner

In this section we present the case, where a BPEL scope is "infected" by a transaction of an external partner. In this case, the first messaging activity in the scope is an activity receiving a message from a partner, which includes a transaction context. The last activity is the only sending a message to the partner in the transaction context. The scope of the partner has to start with the message sending activity and end with the message receipt. Thus, the transaction boundaries are marked by a scope at both partners.

Figure 1 presents an airline booking process and its interaction with a travel agency. To illustrate the examples, we use the Business Process Modeling Notation (BPMN, [27]) as graphical notation for BPEL. If the semantics of BPMN and BPEL are different, we explicitly point that out. Otherwise, the semantics of the BPMN diagram itself and the rendered BPEL process coincidence. This approach is described in detail in [29].

The travel agency is responsible for booking a flight and organizing a visa. In case the visa application is rejected, the activity "Visa Application" fails

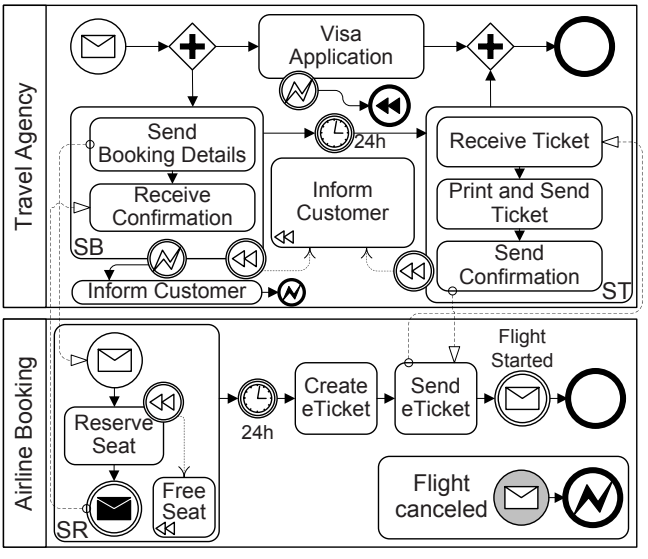

Fig. 1. Airline Booking Process

and throws an error. This error causes the compensation of already completed activities. The airline booking process first receives the booking details, reserves a seat and sends a confirmation to the travel agency. After 24 hours passed, the airline booking creates an eTicket, sends that ticket to the customer and waits for a confirmation. As soon as the flight starts, the process receives a message by the flight management system and the airline booking process ends.

A flight can always be canceled by the airline deposit process. In this case, the airline deposit process sends a flight canceled message to the airline booking process. This message is received by an non-interrupting event. Non-interrupting events are modeled as gray message start events and have BPEL's `onEvent` semantics. The reception of the message is enabled as long as the process is active. The reaction to the flight canceled message must be sending information to the travel agency. This can be achieved in two ways: (a) explicitly sending a message to the travel agency with explicit handling at the travel agency or (b) using the mechanisms of the underlying transaction protocol.

To illustrate the interplay between local transactions and external transactions, we focus on option (b). In this setting, the flight canceled message leads to a fault being thrown. This fault is handled at the process level and leads to the termination of all nested activities and the compensation of all completed activities in reverse order of their completion. The scope SR may be in two states if the fault is raised: (i) running or (ii) completed. In the first case, the BPEL engine needs to terminate the scope which leads to a termination of all nested activities. In addition, the coordination message `Fail` has to be sent to the travel agency. In the second case, the scope completed and sent the coordination message `Completed` to the travel agency. Now, the scope is in the state

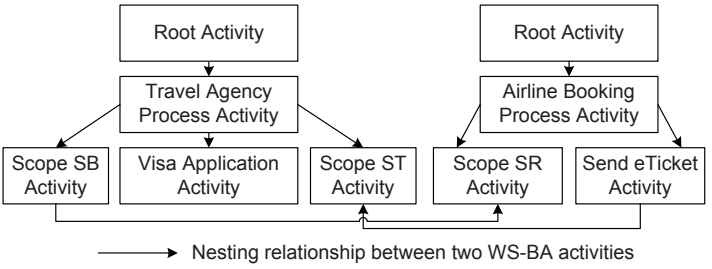

Fig. 2. WS-BA activities and their nesting in the travel agency scenario

"Completed" which enables the travel agency to compensate the scope SB. Due to the fault raised by the flight cancellation, the scope is compensated by the airline booking process. The current standard for compensation-based transactions, WS-Business Activity (WS-BA, [8]), does not allow a participant to compensate if it is in the state "Completed". In our setting, a compensation in the state "Completed" has to lead to a compensation of the other involved transactions, since the effect of one participant has been undone.

By using WS-BA, a transaction context is sent from the "Send Booking Details" activity to the airline booking process [12]. The scope SR is now part of two transactions: The scope is a sub-transaction of scope SR and also a sub-transaction of the airline booking process. Figure 2 presents the nesting graphically. As described in [11], each BPEL process, BPEL scope and invoke activity is a WS-C activity. These WS-C activities are nested in the case that the respective BPEL activities are nested in the process. As shown in [12], a WS-C activity a gets nested into an WS-C activity b if b calls a with a transaction context. Thus, the scope SR is nested in *two* WS-C activities: The WS-C activity of the scope SR and the WS-C activity of the airline booking process.

To enable a compensation in the state "Completed" by a participant, we extend WS-BA with Participant Completion to WS-BA with Participant-Triggered Compensation (WS-BA w/ PTC) as presented in Fig. 3. In WS-BA w/ PTC, the state "Closing" is replaced with a subgraph containing the states "Preparing Closing", "Closing Prepared" and "Closed". These three states are similar

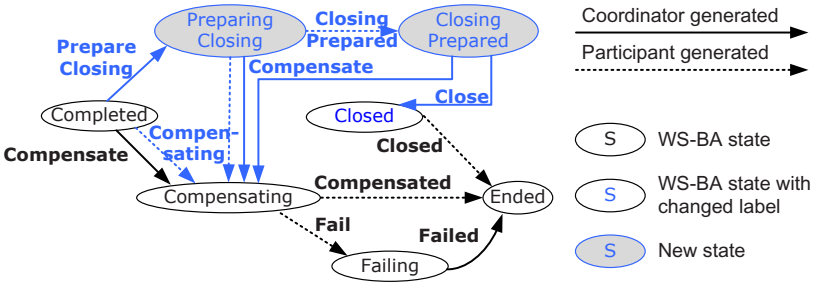

Fig. 3. WS-BA with participant-triggered compensating

to the 2PC protocol, where the participants first vote for the decision and the coordinator finally commits the transaction if all participants voted for commit. Otherwise, a rollback is made at each transaction [30]. In the case of WS-BA w/ PTC, all participants have to be in the state "Completed" to start the closing phase. In the case that not all participants are in the state "Completed" and one participant sends "Compensating" out of the "Completed" state, the transactions of the other participants have to be canceled or compensated: A "Cancel" message is sent to all participants in the state "Active" and a "Compensate" message is sent to all participants in the state "Completed". As soon as all participants are in the state "Completed", the message "Prepare Closing" is sent to all participants. The participants now change to the state "Preparing Closing". From herein, they can either sent "Closing Prepared" or "Compensating". For instance, the latter occurs in the case of the airline booking process: As soon as the scope SR is complete, the flight is canceled and thus the scope SR gets compensated. In case a "Compensating" message is received by the coordinator, it sends the message "Compensate" to all other participants. Otherwise, all participants sent "Closing Prepared" to the coordinator. A participant sends this message in case the transaction cannot be compensated by the participant any more. In case of the airline booking, this happens as soon as the message "flight started" is received and the process reaches its final state. In case all participants are in the state "Closing Prepared", no participant can send a compensation message to the coordinator. Thus, the coordinator can close the transaction by the message "Close". The participant is now in the state "Closed", which equals the state "Closing" of WS-BA. The participant acknowledges the closing by the message "Closed".

In case a fault happens at the activity "reserve seat" (scope SR), this fault has to be communicated to the scope SB. Since the scope SR does not contain a fault handler, the coordination message `Fail` can be directly sent to the coordinator. That message leads to a fault in the scope SB, which in turn leads to informing the customer and a process termination. In case the scope SR had a fault handler, two different messages may be sent to the travel agency process: `CannotComplete` or `Fail`. `CannotComplete` indicates that a fault has occurred and that the fault can be successfully handled by a fault handler. `Fail` indicates that the fault handler did run, but rethrew the fault. The coordinator, however, will be notified about the outcome after the fault handling. It is shown in [11] how WS-BA can be extended to support a notification before the fault handler runs.

The activity "Send eTicket" also starts a transaction. The message containing the eTicket also contains a transaction context. The travel agency receives the eTicket, generates a hardcopy of the ticket and sends it to the customer. Afterwards, the airline booking process receives a conformation that the ticket was handled successfully. In case an error occurs during the printing and sending of the ticket at the travel agency, the transaction protocol ensures that the activity "Send eTicket" raises a fault: The protocol sends `Fault` to the coordinator which in turn sends `Cancel` to the BPEL engine, which in turn raises the fault leading to a compensation of the SR scope and especially a compensation of the

reserve seat activity. By using WS-BA as the underlying coordination protocol, the activity "Send Confirmation" is not necessary any more: The scope ST is a WS-C sub-activity of the WS-C activity for "send eTicket". Thus, the activity "send eTicket" may only complete if the WS-C sub-activity for ST completes.

4 Conclusion and Outlook

This paper presented the situation, where a BPEL scope is part of two transactions. We showed that the coordination protocol WS-Business Activity has to be extended to deal with that case.

In general, the boundary of the transaction is not be equal to the scope boundary. Thus, our ongoing work focuses on defining transaction boundaries using pairs of receive/reply activities, activities accessing the same data item and user defined boundaries. Our future work includes a formal description of the problem as well as a treatment of other coordination protocols such as WS-Atomic Transaction.

Acknowledgment. This work is supported by the BMBF funded project Tools4BPEL (01ISE08B).

References

1. OASIS: Web Services Business Process Execution Language Version 2.0 – OASIS Standard (2007), http://docs.oasis-open.org/wsbpel/2.0/wsbpel-v2.0.html
2. Dayal, U., Hsu, M., Ladin, R.: Business Process Coordination: State of the Art, Trends, and Open Issues. In: VLDB 2001, 27th International Conference on Very Large Data Bases. Morgan Kaufmann, San Francisco (2001)
3. 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), 235–270 (2008)
4. Leymann, F.: Supporting Business Transactions Via Partial Backward Recovery In Workflow Management Systems. In: Datenbanksysteme in Büro, Technik und Wissenschaft, BTW 1995 (1995)
5. OASIS: Web Services Coordination (WS-Coordination) Version 1.2 (2009), http://docs.oasis-open.org/ws-tx/wscoor/2006/06
6. Curbera, F., Leymann, F., Storey, T., Ferguson, D., Weerawarana, S.: Web Services Platform Architecture: SOAP, WSDL, WS-Policy, WS-Addressing, WS-BPEL, WS-Reliable Messaging and More. Prentice Hall PTR, Englewood Cliffs (2005)
7. OASIS: Web Services Atomic Transaction (WS-AtomicTransaction) Version 1.2 (2009), http://docs.oasis-open.org/ws-tx/wsat/2006/06
8. OASIS: Web Services Business Activity (WS-BusinessActivity) Version 1.2 (2009), http://docs.oasis-open.org/ws-tx/wsba/2006/06
9. Khalaf, R., Leymann, F.: Coordination Protocols for Split BPEL Loops and Scopes. Technical Report Computer Science 2007/01, University of Stuttgart (2007)
10. Leymann, F., Pottinger, S.: Rethinking the Coordination Models of WS-Coordination and WS-CF. In: ECOWS 2005 (2005)

11. Pottinger, S., Mietzner, R., Leymann, F.: Coordinate BPEL Scopes and Processes by Extending the WS-Business Activity Framework. In: Meersman, R., Tari, Z. (eds.) CoopIS 2007, Part I. LNCS, vol. 4803, pp. 336–352. Springer, Heidelberg (2007)
12. Tai, S., Khalaf, R., Mikalsen, T.A.: Composition of Coordinated Web Services. In: Jacobsen, H.-A. (ed.) Middleware 2004. LNCS, vol. 3231, pp. 294–310. Springer, Heidelberg (2004)
13. IBM, SAP: WS-BPEL Extension for Sub-processes – BPEL-SPE (2005), http://www.ibm.com/developerworks/library/specification/ws-bpelsubproc/
14. Sauter, P., Melzer, I.: A Comparison of WS-BusinessActivity and BPEL4WS Long-Running Transaction. In: Kommunikation in Verteilten Systemen (KiVS). Informatik aktuell, pp. 115–125. Springer, Heidelberg (2005)
15. Grefen, P.W.P.J., Vonk, J.: A Taxonomy of Transactional Workflow Support. Int. J. Cooperative Inf. Syst. 15(1), 87–118 (2006)
16. Riegen, M.V., Husemann, M., Ritter, N.: Providing Decision Capabilities to Coordinators in Distributed Processes. In: ICIW 2008: Third International Conference on Internet and Web Applications and Services, pp. 500–505 (2008)
17. Coleman, J.: Examining BPEL's Compensation Construct. In: Workshop on Rigorous Engineering of Fault-Tolerant Systems, REFT 2005 (2005)
18. Greenfield, P., Fekete, A., Jang, J., Kuo, D.: Compensation is Not Enough. In: EDOC 2003: 7th International Conference on Enterprise Distributed Object Computing, Washington, DC, USA, p. 232 (2003)
19. Dalal, S., et al.: Coordinating Business Transactions on the Web. IEEE Internet Computing 7(1), 30–39 (2003)
20. Little, M., Webber, J.: Introducing WS-CAF—more than just transactions. Web Serv. J. 3(12), 52–55 (2003)
21. Vetter, T.: Anpassung und Implementierung verschiedener Transaktionsprotokolle auf WS-Coordination. Diploma thesis, University of Stuttgart, IAAS (2006) (in German)
22. UN/CEFACT: UN/CEFACT's Modeling Methodology (UMM), UMM Meta Model – Foundation Module. Technical Specification V1.0 (2006)
23. Hofreiter, B., et al.: Deriving executable BPEL from UMM Business Transactions. In: SCC 2007: IEEE International Conference on Services Computing (2007)
24. Peltz, C.: Web Services Orchestration and Choreography. IEEE Computer 36(10), 46–52 (2003)
25. Decker, G., et al.: Interacting services: from specification to execution. Data & Knowledge Engineering 68(10), 946–972 (2009)
26. Kavantzas, N., et al.: Web Services Choreography Description Language Version 1.0 (2005), http://www.w3.org/TR/ws-cdl-10
27. Object Management Group: Business Process Modeling Notation, V1.2. (2009), http://www.omg.org/spec/BPMN/1.2/PDF
28. Kopp, O., Wieland, M., Leymann, F.: Towards Choreography Transactions. In: 1st Central-European Workshop on Services and their Composition, ZEUS 2009 (2009)
29. Schumm, D., et al.: On Visualizing and Modelling BPEL with BPMN. In: 4th International Workshop on Workflow Management, ICWM 2009 (2009)
30. Gray, J., Reuter, A.: Transaction Processing: concepts and techniques. Morgan Kaufmann, San Francisco (1993)

All links were last followed on June 22, 2009.