External Behavior Modeling Enrichment of Web Services by Transactional Constraints

Ali Khebizi
Computer Science Doctoral School
8 mai 1945 University - BP 401 Guelma 24000 - Algeria
kheali@hotmail.com

Supervised by: Hassina Seridi
Badji Mokhtar University, BP 12 Annaba 23000, Algeria
LabGed laboratory. seridi@lri-annaba.net

Abstract.

The current description of services basing on their interfaces and their protocols remains limited and does not include all the engaged interactions properties in a Web service environment. For a real deployment and a broad adoption of Web services technology, service protocols must be enriched by other properties related to interactions nature between services.
In this paper, we present a model for representing transactional effects and service protocols description improved by injection of the proposed effect model. The protocol compatibility and the equivalence analysis will be reviewed in the light of the proposed enhancements.

Keywords: Web service protocols, Transactional effects, Compensation, Compatibility analysis, Services equivalence.

1 Introduction

The current description of Web services basing on their interfaces and protocols remains limited and does not represent all the semantics of the interactions involved during services invocations. Indeed, current service protocol modeling is taking into account various characteristics which describe its external behavior (such as: order and time constraints) [1][2]. In addition, the current infrastructure of Web services (SOAP, WSDL, and UDDI) is enriched by specifications needed to manage transactions and coordination at Middleware level (such as the Frameworks: WS-Coordination [3] and WS-Transaction [4]). However consideration and management of transactional constraints in services protocols are not given the interest they deserve.
For an actual deployment and broad adoption of this technology, service protocols modeling require other enhancements. In this paper, we propose an enrichment of the external behavior of Web services description by taking into account the transactional constraints. We will also strive to study the conceptual consequences of this enrichment on compatibility and equivalence analysis.
The paper is structured as follows: In section 2, we will explain the motivations of this work. A state of the art of transactions management in service protocols is

presented in section 3. In section 4, we will propose a model for representing transactional effects and we present the involved protocol model. Section 5 will be devoted to analyzing compatibility and equivalence of protocols enriched with transactional effects. Finally, we conclude and present our future works in section 6.

2 Motivations

Because of their excessive cost and their relatively long time, transactions in Web services differ in their effects and their cancellation process from those related to traditional databases. Therefore, service providers are rather compensating transactions and affect often a part of the costs associated to customers [1]. The compensation is provided by the middleware transparently by execution of the compensation protocol predefined by the service developer [5]. In this context, it is appropriate to address a profound reflection on the following issues: How to describe and model transactions effects and their compensation effects? How to model service protocols taking into account the transactional effects? What impacts will bring the injection of transactional constraints in service protocols to the analysis of service compatibility and equivalence?

Only a few studies exist on this issue that requires more in-depth research efforts. In addition to answering previous questions, the following reasons motivate this work.

a. **Services compatibility verification:** the compatibility definition of two protocols (fully or partially), as described in [1], restricts the test criterion to operations order and to messages polarity. It has been extended to take into account the time constraints (time, date) [2]. The compatibility of two services protocols must take into account transactional aspects related to messages and their effects.

b. **To infer transactional properties for existing scenarios:** given BPEL programs availability, it is appropriate to be able to extract their transactional properties for their analysis and their manipulation. In this perspective, a BPEL program is analyzed to extract its transactional properties. Indeed, elements of management errors and transactions, such as: <*compensate*>, <*compensate scopes*> and <*compensation Handler*> constitute activities blocks related to transactional properties that must be recovered and modeled for their possible treatment.

c. Protocol consistency checking (Design Tools): In a compensation situation, it is imperative to check whether the compensation protocol is consistent with the trigger one or not? i.e.: the compensation guarantees – indeed- a "semantic cancellation" of observed effects? Transactions effects modeling and related services protocols management will provide a sound conceptual framework to check the consistency of a protocol for compensation with the trigger one.

3 Transactions management in services protocols state of the art

The state of the art of transactions management -at the protocol level- highlights four approaches that deal in a more or less rigorous way this aspect. In **Protocols languages modeling**, WSFL and XLANG languages provide extensions to the

WSDL standard, offering composition and coordination structures of services based on rules. However, no model is provided for distributed transactions management and transactions compensation is discussed in relation to data flows manipulation. This would require a considerable programming effort. In **Web transactions protocols**, the current Web services specifications are relaxing the ACID properties and strengthening mechanisms for compensation [5]. However, the majority of the proposed specifications don't deal neither with the concept of transactional effect nor with the compensations management. Both protocols have dealt with this issue are: Business Transaction Protocol (BTP) [7] and Tentative Hold Protocol (THP) [8]. In BTP, transactions effects are covered in three dimensions: *provisional effects*, *counter-effects* and *final-effects*. However, specification of effects types remains manual and specific to the engaged coordination. In addition no mechanism to ensure counter-effects consistency with effects is presented. THP is based on the reservation and allocation principle of the current transaction resources by manipulating the concepts: *attempt*, *non-blocking* and *holds reservation*. The cancellation and compensation process are then significantly reduced. But the protocol remains limited in managing the transactions effects and their manipulation. Furthermore, customers have no idea on resources they will need during the activities evolution. **Development environments of business Web services** (Enterprise Java and XML transactions) suffer from the shortcomings due to the lack of conceptual models for representing and manipulating transactional effects. Therefore, no mechanism can verify that observed effects in the real world are, really, those desired apart from traditional testing suites/scenarios. In addition, compensation is discussed in terms of a new process to execute. **The Web Service Transaction model (WSTx)** [9] proposes a WSDL language extension to describe the customer and provider's transactional behavior. However, it suffers from a deficit in modeling effects and proposes only a WSDL operations type classification following transactional criterion.
To address the deficit in effect modeling and compensation management we will propose, in what follows, a formal model for representing transactional effects which is injected thereafter in the service protocol model.

4 Modeling effects and their impact on service protocols model

The Table 1 summarizes models characteristics that bring effects representation on Web services. It presents a comparison on the basis of a set of criteria, highlighting by this their strengths and weaknesses.

Model / Criterion	OWL-S [10]	BPEL	*Colombo* [11]
Concepts	Ontology, Classes, Effects ServiceProfil, ServiceModel,	Activity, Variables, Scope Compensation	Database Query Updating, Universal Relation
Formel Meta Model	Logic description	Language	Relational Model
Concept of effect	Yes	No	Yes
Concept of State	Yes	No	Yes
Compensation handling	No	Yes	No
Formel Model for effects	**No**	**No**	**No**

Table 1: Different models representing effects Comparison

Based on the perception of effects such as query for updating the database, the *Colombo* model [11] offers advantages in effect and state concepts mastery. However, it is still failing in the management of compensating transactions and does not allow comparative effects manipulation.

We will adapt *Colombo* principle model for representing transactional effects. Indeed, in our model, effects and their compensation effects are considered as requests to update the database. Thus, a message of a service protocol will impact on the real world of a customer by executing a request to update database by type: *Insert (R), Delete (R) or Modify (R)*, where *R* is the record of the database reflecting the impact of the message on customer world. The transactional effects managing problem is reduced accordingly to that of handling query, as shown in Table 2.

Transactional effect management problem	**The corresponding Query management Problem**
Checking equivalence and difference of effects	Comparison of updating query
Finding the compensation effect for compensating an	Search a query for cancellation after
Finding elementary effects for complex effect	Queries decomposition
Cumulated effects for a complete execution path	Sequence of queries
Cumulated effects for compensation	Search a query sequence for compensation
Checking transactional effects protocols equivalence	Comparing equivalence of query sequences
Checking transactional effects protocols compatibility	Comparing sequences of query

Table 2: Transformation of effects management problem to updating queries problem

Taking into account transactional effects allows a rich representation of interactions reality in Web services. Indeed, a message will be characterized, in addition to its polarity by effects. It creates in the customer world, as well as compensation effects involved. Compensation effects are represented jointly with observed effects, in order to express the fact that service providers implement charges that differ even if effects are the same. This performance reflects the reality on the diversity of logic compensation which is specific to each provider.

New structure of message for service protocols enriched by the transactional effects: According to our model, at each message is associated a request to update the database and its corresponding complaint related to compensation effects. The new structure of a message is described as follows: ***m (p, e, e')***, where:

m: Refers to the message and its polarity *p (+,-)* as the message is input or output [1]

e: All effects observed in the customer world. This is a request to update the database.

e': All effects of compensation to defeat semantically the effects *e*. This is a request for updating the database to cancel the effects *e* while applying charges imposed by the supplier and relating to the transaction cancellation.

This modeling express in a formal way (relational queries) the effects of transactions and compensation effects for each message of service protocol.

Formal model of service protocols enriched by the transactional effects: Integrating the new structure of the message in the basic model of service protocols [1], will result in an overhaul of protocols model modeled with deterministic finite state machine. The new model protocol (*transactional effects protocol*) is described by the tuple: $\mathbf{P = (S, s_0, F, M, R, S_b)}$ where:

S: A finite set of states; s_0 Є S is the initial state of the protocol; S_b: state of the database associated to each state of protocol;

F: The set of final states machine, with $F \subset S$; M: a finite set of messages, we associate to a message *m* two types of effects *e* and *e*', which correspond, respectively, the requests *Ri* and *Rj* for the database updating.

$R \subset (S \times S_b)^2 \times M$: Transitions set. Each involves a state source, which is associated a database state, to a target state with its database state, following the message receipt. It should be noted: $R((s, s_b),(s', s'_b),m)$ instead of $((s,s_b), (s',s'_b),m) \in R$.

In addition, an effect function is defined, for each message allowing the transition from one state to another (with their database states), combines effects (in terms of requests) and compensations effects (corresponding requests).

5 Transactional effects protocols' compatibility and equivalence analysis

Service protocols compatibility and equivalence analysis as specified in [1] [2] should be revised in the light of proposed enhancements. Indeed, transactional effects representation will be exceeded qualitative analysis, beyond its simple syntax and structural aspect. It will be richer because it is based on semantics of transactions seen in terms of messages effects in the real world. In addition, modeling compensating effects in conjunction with observed effects representing a message by related attributes *(e, e')* expresses perfectly the real situations in which suppliers combine compensation effects for each observed effect.

- ✓ **Transactional effects protocols compatibility:**

Transactional effects protocols compatibility differs from that of basic protocols, due to effects induced by messages. Indeed, two compatible protocols in the basic model [1] may not be in the new context. In this sense, an interaction between two services protocols is allowed only if observed effects will be compatible. By compatibility effects, we are presuming that complaints updating request at the databases have the same type (*Delete, Insert* or *Modify*). This condition implies an interaction path concept redefinition to be extended to query type, as follows:

((State1.State2).Message.QueryType)*

This extension will ensure -when analyzing- verifying the compatibility of updating query and will promote a richer specification of interaction protocols between the candidates. Thus, two service protocols may be compatible only if queries -or sequences of queries- associated to messages would be compliant.

- ✓ **Transactional effects protocols equivalence:**

After studying various scenarios, we concluded that transactional effect protocols equivalence is conditional on final states equivalence of the two databases witch is considered on the basis of sequence equivalence of query updating.

We have identified two equivalence types for queries sequences: strict equivalence and converging equivalence.

Strict Equivalence: For each message *m* of a protocol P_1 corresponds to the corresponding message in the protocol P_2, exactly one query that it is equivalent: *i.e.* it has the same type (*Insert, Delete, and Modify*).

Bases states' converged equivalence: In this case, we are interested in the queries sequence of the complete execution paths. For each query sequence associated to a

complete execution path of a protocol P_1 corresponds to the same path in P_2, another sequence of equivalent queries. This leads to a convergence of updates inducing databases final states which are equivalent.
The two equivalence types induce two equivalence classes for transactional effects protocols: *strict equivalency Class* and *bases states' converged equivalence Class*. The second equivalence class is of particular interest because it expresses a way of achieving differently from the service providers while leading to identical databases.

6 Conclusion

In this paper, we highlighted the interest of transactional constraints modeling. These constraints have been perceived as effects affecting the customer world and have been analyzed in the context of their compensation. We proposed a model based on query for updating databases for representing transactional effects. The enriched service protocol model was presented and formalized. The second contribution is on the compatibility analysis formalization and study of transactional effects service protocols equivalence.
As future work, we plan to identify the compatibility types and to study the algorithmic aspect. We intend, moreover, the proposal for a set of operators handling transactional effects.

References

1. B. Benatallah and al : Representing, Analysing and Managing web Service Protocols. Data Knowledge Ingineering. 58 (3): 327-357, 2006
2. J. Ponge and al: Fine-Grained Compatibility and Replaceability Analysis of Timed Web Service Protocols. ER 2007: 599-614
3. F. Cabrera and al. Web Service coordination (WS-coordination), August 2005
4. F. Cabrera and al. Web Service transaction (WS-transaction), January 2004. http://dev2dev.bea.com/pub/a/2004/01/ws-transaction.html/
5. Gustavo Alonso, Fabio Casati, Hurumi Kuno,Vijay Machiraju : Web services concepts Architectures and applications, Edition Springer Verlag Berlin 2004
6. Web Services Business Process Execution Language Version 2.0 OASIS Standard, 11 April 2007, http://docs.oasis-open.org/wsbpel/2.0/OS/wsbpel-v2.0-OS.html/
7. OASIS Business Transaction Protocol (BTP), http://oasis-open.org/committees/business-transactions/
8. J. Roberts , K. Srinivasan. Tentative hold Protocol Part 1: White Paper, W3C Note 28 November 2001, http://www.w3.org/TR/tenthold-1/
9. T. Mikalsen and al, Transactional attitudes: Reliable composition of autonomous Web Services, WDMS 2002.
10. D. Martin and al, OWL-S: Semantic Markup for Web Services, W3C Submission November 2004, http://www.w3.org/Submission/2004/SUBM-OWL-S-20041122/
11. D. Berardi and al, Automatic composition of Transition-Based Semantic Web Services with Messagging, Proceeding of 31st VLDB conference, Trondheim, Norway, 2005