I.J.Modern Education and Computer Science, **2011, 4, 47-54**
Published Online July 2011 in MECS (http://www.mecs-press.org/)

A framework for ensuring consistency of Web Services Transactions based on WS-BPEL*

Pan Shan-liang,Li Ya-Li, Li Wen-juan
Department of Institute of Computer Science & Technology
Ningbo University
Ningbo, China, 315211
panshanliang@nbu.edu.cn,muwenzijuan520@163.com,

***Abstract*-Transaction processing, as the key technology of web service composition (WSC), has obtained wildly concern. WS-BPEL[1] as a primary web service composition description language, which couldn't coordinate these web service transactions that distribute in a distributed computing environment reach consistent agreement on the outcome. This paper proposed two kinds of transaction types and coordination mechanisms by analyzing the features of WSC transaction, and a transaction processing coordination model based on BPEL was lastly proposed, by which extending the structure of BPEL firstly and then introduced the coordination mechanism into it. The model was validated by an instance at last.**

***Index Terms*-Transaction, service composition, BPEL, coordination mechanism.**

I INTRODUCTION

Major IT organizations such as Amazon, Google, and e-Bay have been migrating their interfaces for business partners to service-oriented architectures using the Web Services (WS) technology. WS allows organizations to easily integrate services across different organizations as well as within organizations. Such WS-based integrated applications should guarantee consistent data manipulation and outcome of business processes running across multiple loosely-coupled organizations. Thus, WS technologies should be extended to equip with transaction-processing functionalities.

There are three proposals for protocols to extend the WS with transaction-processing capabilities, i.e. Web Services Transactions specifications[8], Business Transaction Protocol (BTP) [7], and WS-CAF[9-11]. These specifications provide transaction support in loose coupling environment by relaxing ACID properties of traditional transaction. BTP adopt the two-phase commit protocol to ensure the consistency of the transaction, but it's not very ideal in practice because its centralization management and it doesn't support traditional ACID transaction. The WS-Transaction describes an extensible framework for divorcing the coordination framework and coordination type. But two kind coordination types that are proposed lack of implementation of concrete coordinate process. WS-CAF has certain advantage compare with the former two specifications. It is similar to WS-Transaction, but more completed, however, the WS-CAF has not adopted by any system recently.

WS-BPEL is a mainstream web service composition description language and has become a standard description language. WS-BPEL relied on WSDL is defined as an process-oriented service composition language based on XML ,and is formulated into a norm for web service composition. It can not only implement combination, interaction and presentation process between web services but also WS-BPEL process itself is exposed as a WSDL-defined services and can be called by other web services.

The core concept of WS-BPEL is active, that is a statement or a step in the implementation during the WS-BPEL process.The activity of BPEL is divided into atomic activity and structured activity. The atomic activity such as invoke, reply，receive，assign and so on ,which is used for calling a partner web service and the structured activities such as scope, sequence, flow and so on ,which provide a container for nested activities. The user can combine some web service using these activities. The scope is a special structured activity, which provides context for fault handling and compensation handling, and which provide some support for transaction processing. However, the BPEL couldn't coordinate these web service transactions that distribute in a distributed computing environment reach consistent agreement on the outcome, namely, the BPEL lack of support of transaction coordination mechanism.

This paper proposed a transaction processing model based on BPEL through extending BPEL, which based on analysis of transaction processing mechanism in BPEL and combine with the characteristics of transaction in web service composition environment (WSCE). Firstly, we discuss the transaction in WSCE and divide it into two types, and then raise the coordination mechanism and fault handling methods according to two types of transaction. Secondly, we analyze the shortage of BPEL in transaction processing, extend it and introduce the concept and behavior of transaction into BPEL. Thirdly, the paper proposes the transaction processing coordination model based extending BEPL through introducing the two different kinds of coordination mechanism. Lastly, the model was validated by an employee travel example.

The remainder of this paper is organized as follows. Section Ⅱ discusses the current status and problems of transaction processing. Section Ⅲ put forward the

* This work is supported by: Natural Science Foundation of China （60773072）.

coordination mechanism and algorithm about transaction in WSCE. Section Ⅳ describes the transaction processing mechanism in BPEL and analysis it's the shortage. Section Ⅴ proposes a transaction processing coordination model based on BPEL. An application example is illustrated in sectionⅥ and lastly section Ⅶ concludes the paper.

II Related work

The research of WSC transaction processing can summarized from following aspects.

On the one hand, there are many works study the transactional of WSC based on workflow such as [12]. These works incorporate transaction semantics such as atomicity and isolation to ensure a reliable workflow execution. For example, FLeymann[12] introduced the concept of compensation concept in the IBM FlowMark workflow system to allow the compensation of activities.

On the other hand, there some works about optimization research on transactional of WSC. For example, [13] proposed a new resource coordination algorithm with transaction-aware based on THP transaction model, which improve the success rate of service commit. The [14] define transactional property (such as pivot, compensatable and retriable) for each service, so the service has transaction semantics.

Besides, some scholar study the transaction based on WS-BPEL such as [15, 16]. In [15], Tai et al. used WS- Policy [17] and WS-Policy Attachment[18] to specify the transactional requirements of scopes and partner links. The purpose of the process container and the transaction web service is also to support transactions in BPEL processes. In [16], Wang et al. extended the BPEL4WS to let BPEL4WS specification support transaction through analyzing the requirements of transaction processing in WSC.

In summary, the current transaction processing still exist issues need to be solved as follows. First, the current WSC model and description language does not offer the support of transaction coordination process, and the two-phase commit protocol that adopted by traditional short transaction is not suitable for the long-running transaction. Second, how to deal with the exception when execute the composite service and recover its execution, and how to choose the combination of web services to improve the execution reliability of composite service in the web service composition process.

III Web service composition transaction coordination mechanisms and algorithms

The types of web service composition transaction

WSC is a complicated service which is completed by a series of web service according to certain business logic by cooperation. These web services deploy on the whole internet, with the characters of autonomy, cross-organization, loosely coupled and long running. Due to these natures of web service, the web service transaction couldn't strictly follow the ACID properties. Therefore, this article will divide the web service transaction into atomic transaction and cohesion transaction to meet the requirements of transaction processing in WSCE.

Definition 1: Atomic Transaction (AT) is similar to traditional ACID transaction. It is used to coordinate the short-life operation. The atomic transaction asks the all participants either to commit or to abort having an "all or nothing" property. The AT will lock the resource before it commits. That is to say, the state of transactions cannot be accessed by other concurrent transactions. If Failure, atomic transactions adopt the way of rollback which recover the state of Atomic transaction from execution so as to ensure the consistency of transactional.

Definition 2: Cohesion Transaction (CT) is used to coordinate the long-running distributed transaction. It is unlike AT locking the resource before commit. The participants of CT can commit the transaction by itself, that is to say, the other transaction can access its intermediate state. If fault appear, Cohesion Transaction adopt the way of rollback before submission and the way of compensation after submission to ensure the consistency of transactional. As compensation is semantically cancel the impact of submitted transaction.

Both of AT and CT relaxed the atomicity and isolation of traditional transaction, so web services coordination mechanism would be change when web services interact. We will detail the coordination mechanism and algorithm of AT and CT in the next.

AT coordination mechanism

Atomic transaction is used for short-life operation and it employs 2PC protocol to guarantee the consistency of transaction. Figure 1 is the process of AT coordination.

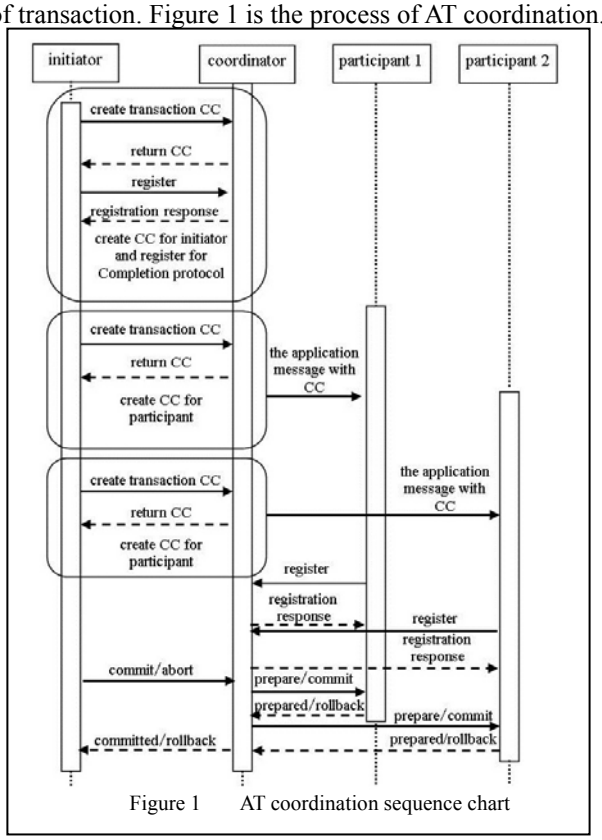

Figure 1 AT coordination sequence chart

(1) Create transaction: The initiator sends the request to coordinator to create CoordinationContext(CC) and ask the participant to join in the transaction. The participant must send a Response message to response it. The coordinator makes a response to express whether it is willing to join the transaction.

(2) Preparation phase: After received the Prepare from coordinator, each participant assign the necessary resource for the execution of its subtask. If success, it send a Prepared to coordinator. If not, send a Not prepared.

(3) Transaction commit: If received N Prepared, the coordinator send Commit to all participants. Otherwise, it will send Abort to each participant to abort the resource allocation. If receive the Commit, the participant will record the commit message in log and execute the corresponding subtask.

(4)Commit failure handling: If any participant send a Failed or the returned messages are less than N, the coordinator will report a failure to user and send Rollback to all participants to recover the committing previous state. If receive N Committed, the transaction complete correctly. Figure 2 is The abstract state diagram of 2PC[21].

(5)Nested transaction: During the execution of the transaction, if some participant itself contains a sub-transaction, it will recursively apply the above mechanism to form the nested transaction tree. At same time participants is not only a participant, but also a coordinate for sub-transaction.

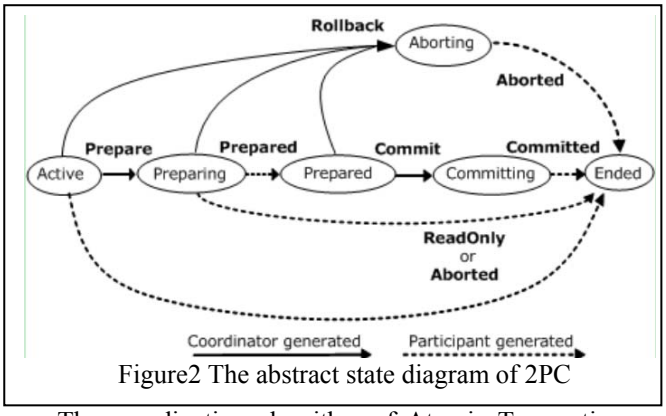

Figure2 The abstract state diagram of 2PC

The coordination algorithm of Atomic Transaction include two parts: coordination algorithm of the coordinator and the participant coordination algorithm. The coordinator and the participant respectively control their own information's transmitting and the communication process to accomplish the whole transaction . The coordinator algorithm and participants algorithm will be provided by the following. The parameter t refers to the waiting time, n1 and n2 express the number of message that the coordinator received, N refer to the number of participant.

The coordination algorithm of coordinator as follows:

```
ActionOfParent{
Step1: initiate a AT
    create atomic coordinator instance and CC;
    send the CC to all participants Pi;
    wait for the response from Pi;
    if timeout exit;
Step2: Prepare for the transaction commit
    send Prepare to all participants;
    while(t ≤ T1)and(n1 < N) wait for and record income messages;
Step3: Commit the transaction
    if(n1=N)and( all n1 messages are prepared){
      record commit in log;
      send Commit to all participants;
        while(t≤T2)and(n2<N) wait for and record income message;
        if(n2<N)and(not N messages are Committed){
          send Rollback message to all participants;
          exit after receiving all Rollbacked;
          exit after receiving Committed;}
        }else {
          send Abort to all participant;
          exit after receiving all Aborted;}
          }
    Participant algorithm as follows:
ActionOfChild{
Step1: join in the transaction;
     creates participant after receiving CC;
    sends Response to Coordinator;
    apply to the coordinator for registration;
Step2: allocate resources
    wait for Prepare from Coordinator;
    if timeout exit;
    success:=allocate resources;
    if( success)send Prepared to coordinator;
    else exit;
Step3: commit sub-transaction
    while(t≤T3)&(message isn't Commit or Abort)
    wait for income messages;
    if(message is Commit){
    record commit in log;
    Commit the sub-transaction;//for nested sub-transaction, call AT coordinator algorithm;
    send Committed to coordinator;}
    else {cancel allocation;
        exit;}
```

CT coordination mechanism

For CT coordination mechanism, this article refers to the interactive processing between coordinator and participant within WS-BA specification. WS-BA is focused on the interaction between coordinator and participant. Compare with the AT coordination process, WS-BA don't define the interface between initiator and coordinator. WS-BA expected workflow engine to provide a proprietary interface to manage WS-BA tasks for the workflow. According to General principles of software architecture，there should be an agreement to determine the interaction between them for different roles. While the tight coupling between the coordinator and the initiator violate this principle.So we need to define an interactive protocol between coordinator and initiator. The WS-BA-I [22] is such a protocol that defines the

interface between initiator and coordinator clearly[23].

During the definition of CT coordination process, It uses WS-BA-I protocol to coordinate the interaction between the initiator and coordinator ,and BAwPC and BAwCC protocol provided by WS-BA to communicate between the coordinator and participants. The BAwCC transition diagram is showed in figure 3.It reacts the changes in the state of transaction when the messages sended by the coordinator and participants receive the appropriate information . The main difference about BawPC and BawCC is that participants can actively commit the transaction if registered BawPC.

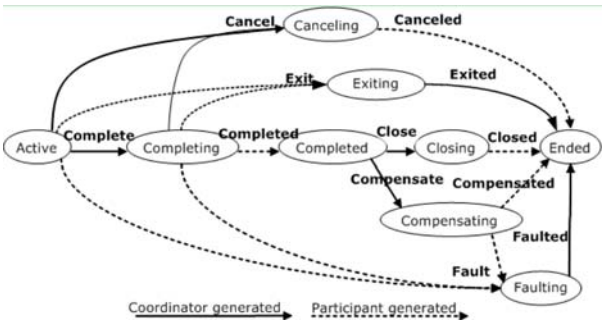

Figure3 The abstract state diagram of BAwCC protocol

It is similar to the process of AT for CT. On the CT coordination mechanism this paper provides with as following: the transaction initiator invokes the activation service which is provided by middleware service to create a CA coordinator and CC. And register for WS-BA-I protocol using registration service according the CC. After registration success, the initiator using the WS-BA-I protocol services to communicate with the transaction's CC. We use the WS-BA protocol to deal with the interaction between coordination and participant. Unlike AT, the participant of CT can commit its own sub-transaction. The specific coordination process is showed in figure 4:

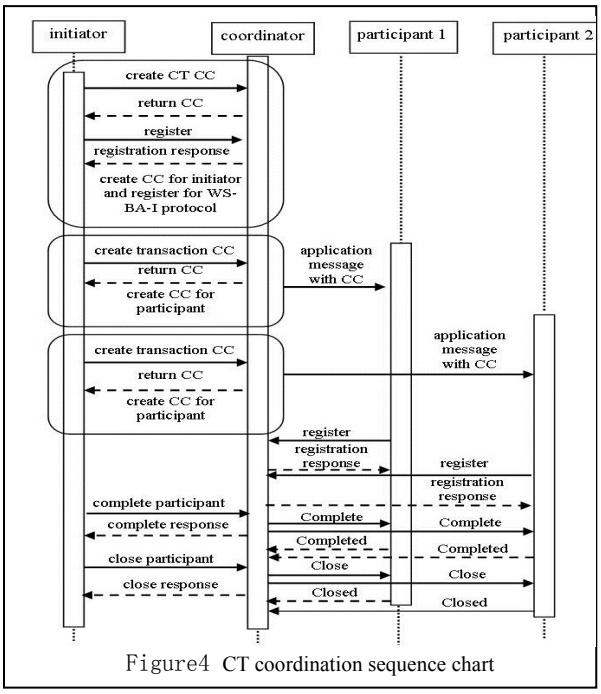

Figure4 CT coordination sequence chart

CT coordination algorithm as follows: The parameter t refer to the waiting time for participants and coordinator.T1 refor to their waiting time threshold. The register protocol of participant is BAwCC. The coordinator support mixed outcome coordination type.

```
ActionOfSuperior{
Step1: initiate a CT
      create a CT transaction
      Create CT coordinator instance and CC;
      while(transaction doesn't complete){
        The coordinator sends CC to participant;
        wait for participant to register ;}
Step2:register/cancel participants
Stpe3:complete/cancel participant
      While(t≤T1){
        wait and record income message;
        if(the initiator select complete some participants){
          send Complete to participants;
          wait for response from participants;}
        else {send Cancel to participant;
              wait the response from participant;}}
Step3:close/compensate participant
      if(participant complete successfully){
        while(t≤T1){wait for incoming message;
          if(the initiator select close some participants){
            send Close to participants;
            wait the response from participants;}
          if(participant need to be compensated ){
            send Compensate to participant ;
            wait for response from participant;}}
}
      Participants algorithm as follows:
ActionOfInferior{
Step1: register to coordinator
        join in the transaction
      create BAwCC participants instance;
      participants apply to the coordinator for registration;
Step2: allocation resource for the participants
      while(t≤T1){wait the response from registration;
          if(register successfully) assign
resource,executive sub-transaction; }
      If(t>T1)exit and get rid of transaction;
Step3:completeclose/compensate sub-transaction
      if(sub-transaction complete successfully){
        while（t≤T1）{
        wait for instruction from coordinator;
          if (message is complete)
            complete sub-transaction,generate
          compensation processing program;
        elseif(message is cancel){
        cancel sub-transaction,release allocation;}
        Step4:close / compensate sub-transaction
         if (sub-transaction complete successfully){
        while(t≤T){
        wait for the instructions from coordinator;
        if(message is close)
          exit and send Closed to coordinator;
        elseif(message is Compensate)
```


{execute the compensate transaction;
if(compensate successfully)
send Compensated to coordinator and exit;
elseif(fault happen) { send Faulted to coordinator ;
cancel allocation ;
exit;} }}

IV THE TRANSACTION PROCESSING MECHANISM AND SHORTAGE IN BPEL

WS-BPEL is a business process description language based on XML, which provides an approach that normally descript the business process and business interaction protocols. WS-BPEL provides scope, fault handling and compensation handling to support transaction processing. But these supports can not satisfy the WSCE. This section will discuss the transaction processing mechanism in WS-BPEL and analysis the BPEL's weakness in WSCE.

The WS-BEPL's transaction processing mechanism

The WS-BPEL as a primary web service composition description language, whose support of transaction processing is the key to the widely use of WSC. The transaction within WS-BPEL mainly concentrates upon scope which encloses a series of activities to complete a certain function. And the scope allows nested. There is a certain similarity with the transaction at this point. Through the fault handler defined by itselves and compensation handler, Scope handle the exception or error occurred in the process of Composite service operation.Figure 5 shows the structure of the scope. A scope can define a compensation handling and multiple fault handling.

The compensation handler is the core of long-running transaction within WS-BPEL [49]. When composite services operate,if something going wrong，it would be necessary to cancel part of the completed operation so as to recover from the implementation of operation .At this time , the provided Compensation handler that semantically cancel the impact of completed operation should be invoking. In a running process, there would be only one time to invoke the compensation handler for a completed scope, or the composite services operation will go wrong.

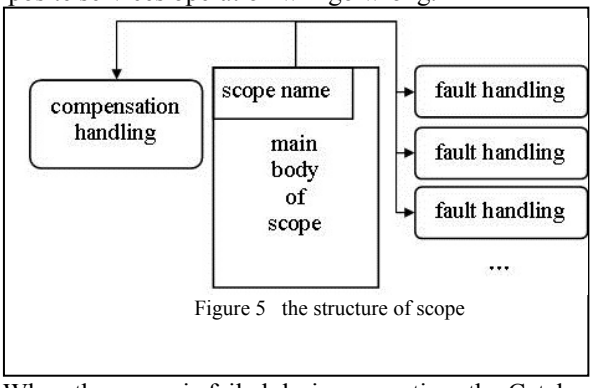

Figure 5 the structure of scope

When the scope is failed during operation, the Catch activities in fault handler will catch the error to respectively invoke the corresponding fault handler by Classifying captured error.

The BEPL's shortage in transaction processing

Through the WS-BPEL business processes, Combination services establish two ways to support the transaction: First, the combined services calls for a single web service as a participant in the process; Second, the process will be packaged into a web services which can be called by other combinations as a transaction participants. When a combination web service running, that is, a transaction is started, It needs to trigger the web service or composite service to register as participants by themselves[25] . During the process, It involves the creation of the transaction, transmission and reception of coordination context , so WS-BPEL is required to have these behaviors. But there are no concepts of transaction in WS-BPEL which lack the mechanism that coordinate multiply participant reach an agreement outcome in WSCE. That is to say, the WS-BPEL lacks the support of transaction coordination mechanism. So we extend the WS-BPEL language for introducing the transaction coordination mechanism

V THE EXTENDING MODEL OF BPEL TRANSACTION

WS-BPEL can enclose a series of sub activities in a structure activity to complete some function, so we can view the structure activity as the transaction boundary. That is to say, we should define transactional behavior for the structure active. But, for the short-life AT and long-running CT based on compensation mechanism, since their coordination mechanisms are different, we should specify different transactional behavior for each. For example, define a transactional scope activity with ACID properties, so the nested activities within scope (such as invoke) will use the 2PC protocol of WS-AT to commit.

The following will give a detailed description on WS-BPEL extending. It's include the introducing of the transaction property, the extending of AT and CT.

A *The transactional extending of BPEL*

1) The main construct for handling a Business Transaction/Business Activity within WS-BPEL is a CC. This is a value that represents a business transaction and is held in a variable, appropriately typed in the variables element:

<variables>
…
<variable name="employeeStatusContext"
type="wscoor:CoordinationContext"/>
</variables>

2) Constructs are needed to specify how CC are transmitted and received with the application messages sent and received in BPEL. There are three constructs that interact with external partner, namely invoke, receive and reply.

receive: This construct is mainly used to start the

business process by receiving external invocation. The receive activity will gain CC when receive external invocation. So we add a property of businessTransactionContext for receive activity.

<receive partnerLink=”AmericanAirlines”
portType=” tns:FlightAvailabilityPT”
operation=” FlightAvailability”
variable=” FlightDetails”
businessTransactionContext=”
AmericanAirlineContext”>
…
</receive>

invoke: This construct is mainly used to call a web service provided by a partner to complete business operation. We extend two properties of inputBusinessTransactionContext that expresses the external CC and outputBusinessTransactionContext that expresses the current CC for invoke activity.

<invoke partnerLink=”employeeTravelStatus”
portType=”emp:EmployeeTravelStatusPT”
operation=” EmployeeTravelStatus”
inputVariable=” EmployeeTravelStatusRequest”
inputBusinessTransactionContext=”travelTransactio
nContext”
outputVariable=”EmployeeTravelStatusResponse”
outputBusinessTransactionContext=”employeeStatu
sContext”>
…
</invoke>

reply: This construct is mainly used to respond the external invocation in the synchronization interactive process. It needs to return the CC to its partner, so we add an outputBusinessTransactionContext property that use to return the CC to its partner service.

<reply partnerLink=”employeeTravelStatus”
portType=”tns:EmployeeTravelStatusPT”
operation=”EmployeeTravelStatus”
variable=”EmployeeTravelStatusResponse”
outputBusinessTransactionContext=”
employeeStatusContext”>
…
</reply>

3) Constructs are needed to specify how a business transaction is initiated. This will cause the creation of a new, propagatable CoordinationContext in BPEL. So we extend a new data structure businessTransaction in BPEL, which has the properties of name, action, and context. The name express the name of transaction, the action express the behavior of transaction (such as begin, prepare, complete, compensate and so on) and context express the CC of transaction.

<businessTransaction action=”new”
context=” travelTransactionContext” />
<businessTransaction action=”confirm”
context=”travelTransactionContext”/>
<businessTransaction action=”prepare”
context=” employeeStatusContext”>
<businessTransaction action=”cancel”
context=” employeeStatusContext”>

The atomic activity transactional processing

The WS-BPEL doesn’t provide the support for AT. According to AT coordination mechanism, the transaction must rollback if the transaction executes fail. That is to say, the BPEL process can send a rollback to coordinator of transaction immediately. Figure 6 shows a process with an atomic scope activity that contains a sequence with two invoke activities, interacting with two different partners.

1) When the scope is started, the process tells the middleware to create an atomic CC using the activation service.

2) The activation service returns an atomic CC, which will be sent with the application messages of the nested invoke activities.

3) The AT CC tells the partners (that must support WS-AT) to register at the registration for the 2PC protocol.

4) After the scope activity completes, the BPEL process registers for the completion protocol at the registration service.

5) The BPEL initiates the completion protocol and tells the coordinator to commit.

6) The latter runs the 2PC protocol with the partners A and B and send the result to the BPEL process using the completion protocol.

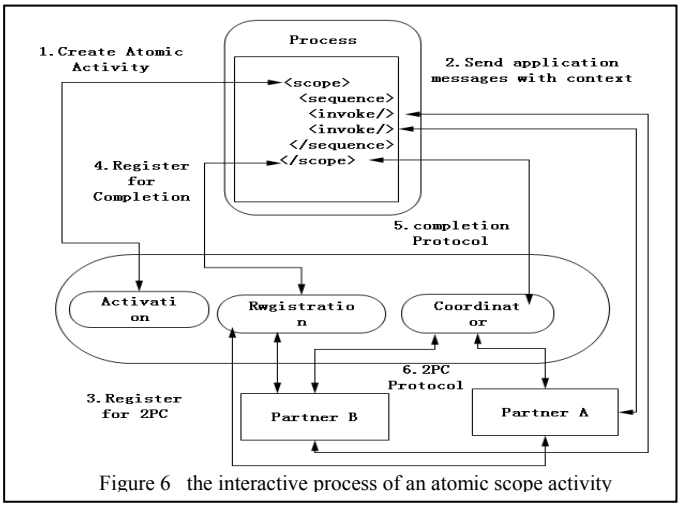

Figure 6 the interactive process of an atomic scope activity

In the process of atomic activities, the process tells the coordinator whether the transaction should be committed or rolled back. This means that the BPEL process or the BPEL engine must be able to perform the following actions: creating a CC, registering for completion protocol, adding a CC to application messages, and supporting the completion protocol. The processes of CT are similar to AT, which are not discussed here.

The fault handling of transaction

If the transaction activity fails, the transaction coordinator should be told to rollback/compensate because the activity will not complete. To support

rollback, we must add a fault handler to the BPEL process that calls the appropriate rollback/compensate operation of the transaction middleware.

Usually, the transaction' state use variable to save. In order to restore the transaction' state, we need to add a new variable to save the variable values before the execution of a transaction. So, if the transaction needs to roll back, we only used this new variable to restore the variable that saves the state of transaction.

For AT and uncompleted CT, we used rollback to keep their consistency. For completed CT, we adopt compensation to keep the consistency. The compensation is that undo the effects of committed transaction.

VI Application Example

Let us illustrate the process by an example of Employee travel arrangements. There are the definitions of simplified business processes of employee travel arrangements: Customers call this process, specify the employee name, destination, departure date and return date. The long-running transaction involves two participants: one service is to arrange an employee; the other service is to schedule flights. It is considered as a successful implementation of the transaction once both of them succeed. If there is a failure between the two, you need to apply the successful service to make a compensation to ensure the consistency of the state of the transaction. Let's detail the two services:

It implements two operations during the process of arranging employees Services, one is to query the status of employee information (whether the employees could be arranged), and return flight standards of employees (may be economy class, business class or first class); the other is to set the status of employee (if employee status could be arranged, then automatically set the operation).

It also implements two operations in the service of booking airline tickets. One is the inquiry services (to query whether there are flights to meet the conditions on the basis of the information provided by customer and the employees flight standards returned by employee services); the other is the scheduled services (the user begin to book after selecting the appropriate flight).

In the combine services, invoking a web service will trigger the web service to automatically register as a participant, so web services can provide a specific operation as a participant (such as submission, completed, etc.). In addition, in this paper, we firstly define describing documents WSDL of web service, by WSDL2JAVA tools, to create participants' service operation interface. But the specific interface does not provide any operation, so participants need to implement the specific operation of participant service in this interface. Only in this way can we complete a participants' service process. These consist of arranging employee services and scheduling flight service.

When participants service successfully released, we can use WS-BPEL language to logic combine published services together according to business Process . As shown in Figure 7.

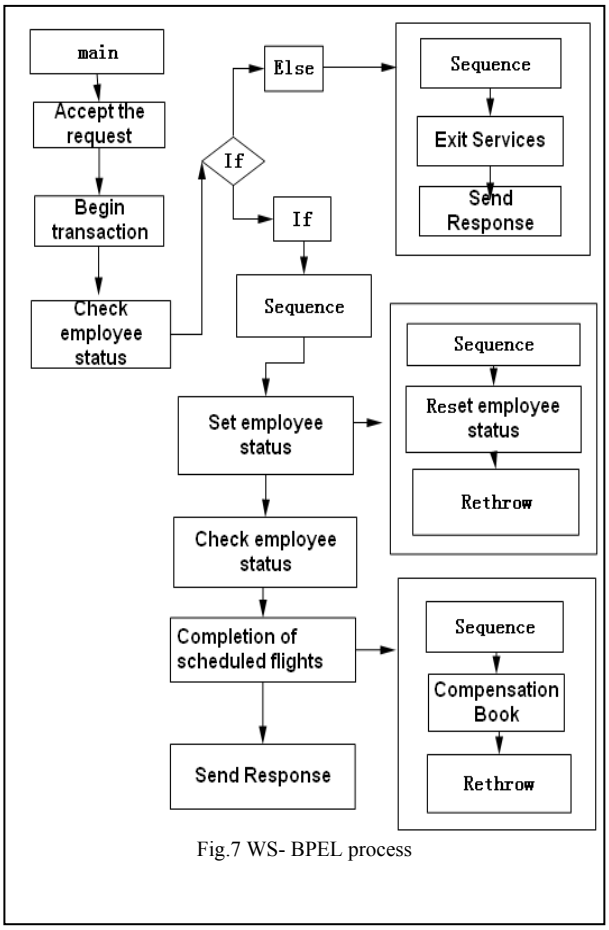

Fig.7 WS- BPEL process

Specific operational procedures are as follows:

1) The user enters the employee's name and travel dates, click Start, then the composite service start to run, the transaction started.

2) Arranged under the names of the employees call the state of employee services available employee operation, if the state is busy, then exit the transaction. If the idle state, then return the employees flight standards and invoke the operation to set the state of employees, and set the state of employees busy in the database.

3) According to the user input travel dates and returned standards of flight, it will call querying flight information of scheduled aircraft flight services. If it meets the conditions for flight information, it will return the relevant information and subtract the number of remain flights within the database. If there are not flight information to meet the conditions ,then returned nothing.

4) According to the returned flight information, the user choose the right flights and call the operations to complete flight scheduling , add the unselected the number of remain flights within the database ,at the same time ,return success reservation message, the transaction ends.

5) If the scheduled operation fails, then it will call compensation operations, the number of tickets plus 1, and forward operation will be called to reset the state of employee, and return transaction failure information.

VII Conclusion

With the rapid development of web application and related technology, the WSC has attained wildly attention. But the transaction processing is the key to the development of WSC, which determine whether the WSC can be wildly used. The research of web service transaction becomes a key and urgent issue within WSC. Although the WS-BPEL is a primary web service composition description language, it doesn't provide enough the support for transaction processing.

This paper design and propose a transaction coordination model base extending WS-BPEL. The mode can coordinate multiply service reach consistent agreement on the outcome. Comply with the original WS-BEPL transaction processing, our model can guarantee the consistent of composition service's execution.

However, this model isn't validated in mathematics, so we will adopt the mathematical tools of Petri net to validate the deadlock-free and accessibility of this model in our next experiment. In addition, this model is only a preliminary model, in order to make the success rate of execution of WSC higher, we will introduce THP protocol into this model to make it more perfect.

Reference

[1] OASIS. Business Process Execution Language for Web Service(WS-BPEL)[EB/OL]. Version 2.0, 2007-05

[2] J.E.B. Moss.Nested Transactions: An Approach to Reliable Distributed Computing [M]. Massachusetts Institute of Technology.1981.

[3] C.Mohan. Tutorial: Advanced Transaction Models - Survey and Critique[C]. In Proceedings of the 1994 ACM SIGMOD international conference on Management of data, 521-521, 1994.

[4] G.Weikum, H.J.Schek. Concepts and Applications of Multilevel Transactions and Open Nested Transactions [M]. Database Transaction Models for Advanced Applications, ed. A.K. Elmagarmid, Morgan Kaufmann, 1992.

[5] H. Garcia. Molina, K.Salem. SAGAS [M]. ACM SIGMOD Record, 1987, 16(3):249-259.

[6] Benchaphon Limthanmaphon, Yanchun Zhang. Web Service Composition Transaction Management[C]. In Proceedingf of the 15th Australasian Database Conference, 171-179, 2004.

[7] OASIS. Business Transaction Protocol (BTP) Version 1.0 4[EB/OL].June 2002. http://www.oasis-open.org/committees /download.php/1184/2002-06-03.BTP_cttee_spec_1.0.pdf

[8] Cabrera, F., Copeland, G., Cox, B., Freund, T., Klein, J., Storey, T., and Thatte, S. (2002) 'Web Services Transaction (WS-Transaction)' http://www-106.ibm.com/developerworks/library/ws-transpec/

[9] Aljuna, Fujitsu, IONA, et al.Web Services Composite Application Framework(WS-CAF).Version 1.0 2003-07

[10] Aljuna, Fujitsu, IONA, et al.Web Services Context(WS-Context).Version1.0, 2003-07

[11] Aljuna, Fujitsu, IONA, et al. Web Service Coordination Framework(WS-CF).Version 1.0,2003-07

[12] F.Leymann. Supporting business transactions via partial backward recovery in workflow management systems. In Proc.of BTW, 1995.

[13] Xu Wei , Chen Wen-qing , Li Bing. taTHP: an Improved Transaction Model Based on THP [J]. Journal of Chinese Computer Systems, 2007, Vol.28 No.1

[14] Sami Bhiri, Claude Godart,Olivier Perrin. Reliable Web Services Composition using a Transactional Approach. IEEE International Conference on e-Technology, e-Commerce and e-Service, 2005

[15] Wang Yong , Zhang Yu , Yin Rui. Research of Business Transaction Process in Web Service Compostion[J]. Journal of Chinese Computer System, 2006, 27(1):121~125

[16] Jeffry. Schlimmer (Eds.). Web Services Policy Framework (WS-Policy). ftp://www6.software.ibm.com/software/developer/ library/ws-policy.pdf , September 2004.

[17] Chris Sharp (Eds.). Web Services Policy Attachment (WS-PolicyAttachment). ftp://www6.software.ibm.com/software/ developer/library/ws-polat.pdf, September 2004.

[18] S.Tai, R.Khalaf, T.Mikalsen .Composition of Coordinated Web Services .In Proc.of the 5th International Middleware Conference(Middleware) , volume3231 of LNCS, pages 294–310.Springer, October 2004.

[19] Jiang-Hua Li,Hui-Qiong Zeng,Song-Jiao Chen. Combination of QoS-constrained service transaction recovery algorithm[J], Computer Engineering, 2008,34(14):41-46.

[20] Yi Ren,Quan-Yuan Wu,Wei-Hong Han,Jiang-Bo Guan, Transaction Processing Technology Review[J], Computer Research and Development, 2005,42(10):1779-1784.

[21] Mark Little, Andrew Wilkinson.Web Services Atomic Transaction (WS-AtomicTransaction), Version 1.1[EB/OL]. http://docs.oasis-open.org/ws-tx/wstx-wsat-1.1-spec-errata-os/wstx-wsat-1.1-spec-errata-os.html, 2007.

[22] Hannes Erven, Georg Hicker, Christian Huemer, Marco Zaptletal. The Web Service-BusinessActivity- Initiator(WS-BA-I) Protocol:an Extension to the Web Service-BusinessActivity Specification [C].IEEE International Conference on Web Services, Washington DC:IEEE Computer Society,2007:216~224

[23] Chun-Hua Yuan,Bo Chen,Ming-Tian Zhou, WS-T protocol suite and its application[J], Computer Applications, 2008, 25(9): 2798-2800.

[24] Bo-Liu Jia-Ju Wu, Web Service Composition Model for Distributed Coordination[J], Microelectronics and Computer, 2006, 23(10): 207-210

[25] Mietzner R. Extraction of WS-BA from BPEL 1.1[D]. Stuttgart:University of Stuttgart, 2006.

