CHAPTER 4

ABSTRACTION FOR THE SPECIFICATION OF
TRANSACTIONAL PROPERTIES

In order to achieve failure tolerant or reliable execution of a composite service, each of the component services must be selected with appropriate transactional capabilities so that the desired recovery mechanism can be applied in case of its failure during execution. Such transaction aware selection of services requires transactional capabilities or properties of a service to be specified in its service description. In addition, the desired transactional properties of services must be obtained from business analyst and matched with the transactional capabilities of the advertised services in the service registry in order to select suitable ones. Most of the existing works on specification of transactional properties require the business analyst to write rules or dependencies which is a difficult task. However, it is very difficult for business analysts to visualize or map the business policies as transactional properties of the services and hence, it is essential to enable them with an abstraction that is simple and flexible.

Towards this objective, the recovery costs of services with different transactional properties are derived in subsection 4.1. In subsection 4.2, a concept of recoverability for web services is proposed, that abstracts the transactional requirements that enables proper grouping of services under different recoverability levels. Subsection 4.3 presents the empirical results to validate the gradation of recoverability levels. Lastly, subsection 4.4 summarizes the proposed approach.
4.1 ESTIMATION OF RECOVERY COST

A composite service comprises of several component services composed in sequence or in parallel. Let us assume that a composite service has \( n \) components and consider the failure of \( i^{th} \) service. In general, the compensation logic of a compensatable service does not incur more time compared to its normal execution time. Similarly, the cancellation logic of a cancelable service consumes lesser time than its normal execution time. It is assumed that there is no failure during the execution of compensation or cancellation logic. The pattern in which the services are composed has an impact on the recovery time of the failed service. Hence, the recovery time of a failed service is computed to be the larger of the recovery time incurred when it is sequentially composed and the recovery time incurred when it is composed in parallel. In order to compute the recovery cost of the failure or interruption of an instance of \( i^{th} \) service, the following terminologies and assumptions tabulated in Table 4.1 are used.

<table>
<thead>
<tr>
<th>Notation</th>
<th>Explanation</th>
</tr>
</thead>
<tbody>
<tr>
<td>( T_r^i )</td>
<td>Execution time incurred by ( i^{th} ) service</td>
</tr>
<tr>
<td>( N_{SC}^i )</td>
<td>Number of successfully completed services</td>
</tr>
<tr>
<td>( 0 &lt; a_i &lt; 1 )</td>
<td>Amount of work completed before a failure or cancellation</td>
</tr>
<tr>
<td>( 0 &lt; b_i \leq 1 )</td>
<td>Effort required for compensating the service expressed as a percentage of its execution time</td>
</tr>
<tr>
<td>( 0 &lt; c_i &lt; 1 )</td>
<td>Effort required for cancelling the service expressed as a percentage of its compensation time</td>
</tr>
<tr>
<td>( T_{cp}^i = b_i \ast T_r^i )</td>
<td>Compensation time</td>
</tr>
<tr>
<td>( T_{cc}^i = c_i \ast T_{cp}^i )</td>
<td>Cancellation time</td>
</tr>
<tr>
<td>( \text{rCount} )</td>
<td>Number of retries before the ( i^{th} ) service succeeds</td>
</tr>
</tbody>
</table>
4.1.1 Backward Recovery Cost

In the event of the failure of an instance of $i^{th}$ service that is sequentially composed in a composite service, all $i$-1 services that have been previously completed must be compensated. The compensation recovery time required for recovering from the failure of a non-retrievable service $i$, $RT_{cp}^i$, is computed to be the maximum of the recovery time incurred when it is sequentially composed $RT_{cpSeq}^i$ and the recovery time incurred when it is composed in parallel $RT_{cpPar}^i$.

$$RT_{cp}^i = \max( RT_{cpSeq}^i , RT_{cpPar}^i ) \quad (4.1)$$

The recovery time incurred by the failure of a non-retrievable service that is sequentially composed, $RT_{cpSeq}^i$ is given below.

$$RT_{cpSeq}^i = \sum_{j=1}^{N_{seq}} T_{cp}^j \quad (4.2)$$

When a non-retrievable service is composed in parallel with $P$ component services, its failure triggers compensating all the completed parallel services and cancelling all the active concurrent services. Then, all the $(i-1)$ services that have been completed prior to the $i^{th}$ service that failed are compensated. The recovery time incurred by the failure of a non-retrievable service that is composed in parallel, $RT_{cpPar}^i$ is lesser when some of the concurrent services have completed while some others are active compared to when all of the concurrent services have completed. Hence, $RT_{cpPar}^i$ is computed to be the maximum of recovery time incurred when all the parallel processes have completed their execution and recovery time incurred when all of them are active. The compensation or cancellation of concurrent processes
occurs simultaneously. Hence, the compensation or cancellation time of all completed or active concurrent services is taken to be the largest of all.

\[ RT_{cp}^i = \max \left[ (\max_{1 \leq k \leq p} T_{cp}^k) + \sum_{j=1}^{N_{sc}^i} T_{cp}^j \right], \]

\[ (\max_{1 \leq k \leq p} T_{cc}^k) + \sum_{j=1}^{N_{sc}^i} T_{cp}^j \]  \hspace{1cm} (4.3)

The recovery of a failed non-retrievable service, when the concurrently running services are active, involves compensating their completed portions. Alternatively, it involves entirely compensating all the concurrently running services when they all have completed their execution. Thus, the recovery time of a compensatable service that is concurrently composed is given by

\[ RT_{cp}^i = \max_{1 \leq k \leq p} (T_{cp}^k) + \sum_{j=1}^{N_{sc}^i} T_{cp}^j \]  \hspace{1cm} (4.4)

The recovery of a failed non-retrievable service that is composed in parallel involves compensating the concurrent services in parallel, in addition to compensating the previously completed services. Alternatively, in case of the failure of a sequentially composed compensatable service, the recovery involves compensating all the previously completed services in sequence in the reverse order of their execution. Therefore, the recovery time to restore from the failure of a non-retrievable service, \( RT_{cp}^i \) given in Equation (4.1) is derived from Equations (4.2) and (4.4) as below.

\[ RT_{cp}^i = \sum_{j=1}^{N_{sc}^i} T_{cp}^j \]  \hspace{1cm} (4.5)

### 4.1.2 Cancellation Recovery Cost

When an instance of the \( i^{th} \) service that is long-running and sequentially composed is interrupted due to changes in input requirements, in
addition to terminating its execution and compensating its completed portions, all the previously completed \((i-1)\) services are compensated. The recovery time of an interrupted cancelable service \(i\), \(RT_{cc}^{i}\) is computed to be the maximum of the recovery time incurred when it is sequentially composed, \(RT_{ccSeq}^{i}\) and the recovery time incurred when it is composed in parallel, \(RT_{ccPar}^{i}\).

\[
RT_{cc}^{i} = \max ( RT_{ccSeq}^{i}, RT_{ccPar}^{i} ) \tag{4.6}
\]

The recovery time incurred by the interruption of a cancelable service that is sequentially composed, \(RT_{ccSeq}^{i}\) is given below.

\[
RT_{ccSeq}^{i} = T_{cc}^i + \sum_{j=1}^{N_{Sc}^{j}} T_{cp}^j \tag{4.7}
\]

The cancellation of a cancelable service that is composed in parallel with \(P\) component services triggers compensating its completed portions and compensating completed concurrent services or cancelling the active concurrent services. Then all the services that have completed prior to the interrupted service are compensated. The recovery time incurred by the cancellation of a cancelable service that is composed in parallel, \(RT_{ccPar}^{i}\) is computed to be the maximum of recovery time incurred when all the parallel processes have completed their execution and recovery time incurred when all of them are active.

\[
RT_{ccPar}^{i} = \max \left[ (\max ( T_{cc}^i, \max_{1 \leq k \leq P} ( T_{cp}^k ) + \sum_{j=1}^{N_{Sc}^{j}-P} T_{cp}^j ) ),
(\max_{1 \leq k \leq P+1} ( T_{cc}^k + \sum_{j=1}^{N_{Sc}^{j}} T_{cp}^j ) ) \right] \tag{4.8}
\]

The recovery of the cancelled service, when the concurrently running services are active, involves compensating only the completed portions of
concurrent services. Alternatively, the recovery of a cancelled service, when the concurrently running services have completed their execution, involves entirely compensating them. Thus, the recovery time of a cancelable service that is concurrently composed is given by

\[
RT_{ecPar}^i = \max(T_{ec}^i, \max_{1 \leq k \leq p} (T_{cp}^k)) + \sum_{j=1}^{N_{ec}^j - P} T_{cp}^j
\]  

(4.9)

The recovery of a cancelled service involves compensating its completed portions. In addition, the recovery of a cancelled service that is composed in parallel involves compensating the concurrent services in parallel and compensating the previously completed services. Alternatively, in case of the failure of a sequentially composed cancelable service, the recovery involves compensating all the previously completed services in sequence in the reverse order of their execution. Therefore, the recovery time of a cancelled service \( RT_{ec}^i \) given in Equation (4.6) is derived from Equations (4.7) and (4.9) as below.

\[
RT_{ec}^i = T_{ec}^i + \sum_{j=1}^{N_{sc}^j} T_{cp}^j
\]  

(4.10)

### 4.1.3 Forward Recovery Cost

When an instance of the \( i^{th} \) service that is retriable fails, it is invoked multiple times until it succeeds. For every failure, the time incurred is lesser than its actual execution time. The failure is restored to a consistent state by successfully executing the service at the last retrial. The recovery cost is the same in cases of the retriable service composed in sequence as well as in parallel. The recovery time, \( RT_{r}^i \) of \( i^{th} \) service that is retriable, is computed as below.
\[ RT_r^i = \left( \sum_{j=1}^{\text{Count}_r-1} a_j \cdot T_e^j \right) + T_e^i \]  

(4.11)

4.2 RECOVERABILITY LEVELS OF TRANSACTIONAL WEB SERVICES

The proposed concept of recoverability levels pertaining to transactional web services abstracts their transactional requirements. The eight kinds of transactional web services discussed in Chapter 3 must be ordered based on their recovery cost so that the services with an optimal cost and required recoverability levels can be selected.

In a composite service, the failure of a pivot component service never leaves any trace which means that it does not leave the execution of composition in an inconsistent state. The execution of the composite service can proceed with subsequent component services, ignoring the failure of the pivot service. Therefore, failure of a pivot service does not require any type of recovery. In other words, a pivot service does not enable any kind of recovery mechanism. Its recoverability level is low and is considered to be one.

In comparison with compensation recovery cost, the cancellation recovery of \( i^{th} \) service involves an additional cost of cancelling the service execution and compensating completed portions of the cancelled service \( i \) (see Equations (4.5) and (4.10)). The cancellation recovery cost is found to be higher than the backward recovery cost. Hence, services with compensatable property are grouped in recoverability level two and the services with cancelable property are grouped in level three. The recovery cost of a service with transactional properties compensatable and cancelable \( \{cpcc\} \) is the larger of its compensation cost and cancellation cost which means it incurs the same cost as a service with property \( \{cc\} \). Hence, such services are also grouped in recoverability level three.
In the event of failure of an $i^{th}$ service that is not retriable, all the previously completed services need to be compensated in order to recover to a consistent state. If the failed service is retriable, then the number of times the failed service or an alternate service is reattempted before it succeeds determines the execution effort in recovering the $i^{th}$ service. It succeeds in the last reattempt. Since the service may fail any time during its execution, the average execution effort in $n$ reattempts will be $\frac{1}{2} T^i_e$. The number of reattempts plays a vital role in comparing the backward or cancellation costs with the forward recovery costs when $i^{th}$ service fails. The number of reattempts up to which the cancellation cost is equal or more than the forward recovery cost is given below:

$$
\text{rCount}_{ccMin} = \left\lfloor \frac{RT^i_{sc}}{\frac{1}{2} T^i_e} \right\rfloor
$$

(4.12)

The number of reattempts up to which the compensation cost is equal or more than the forward recovery cost is computed as below:

$$
\text{rCount}_{cpMin} = \left\lfloor \frac{RT^i_{sp}}{\frac{1}{2} T^i_e} \right\rfloor - 1
$$

(4.13)

The forward recovery cost exceeds backward (cancellation) recovery cost if the failed service is reattempted beyond $\text{rCount}_{cpMin}$ ($\text{rCount}_{ccMin}$) times. In general, it is desirable to retry a failed service until it succeeds to recover from transient failures, rather than undoing the previously completed services. In the worst case, the retriable failed service may be
reattempted more than $r\text{Count}_{cp\text{Min}}$ times i.e. $r\text{Count} > r\text{Count}_{cp\text{Min}} (r\text{Count} > r\text{Count}_{cc\text{Min}})$ until it succeeds, in which case forward recovery cost is higher than the backward (cancellation) recovery cost. The cancellation cost was proved to be more than the compensation cost. Hence, services with property \{r\} are grouped in recoverability level four for which the cost involved in recovery is higher than that of the non-retrievable cancelable services.

A service with \{cpr\} property may succeed after $r\text{Count}$ retries and may be compensated later. While comparing a service with transactional property \{cpr\} and that with \{ccr\}, the recovery cost of \{ccr\} services is higher as cancellation is more expensive than compensation. Hence, services with \{cpr\} property are grouped in recoverability level four which is the same as that of services with property \{r\}. Alternatively, services with \{ccr\} property are grouped in a higher recoverability level of five. A service which has compensatable, retrievable, and cancelable properties may get cancelled or compensated after $r\text{Count}$ retries and its recovery cost is the larger of that of a service with \{ccr\} property and a service with \{cpr\} property. Hence, the recovery cost of services with property \{cpccr\} are the same as that of services with \{ccr\} property and such services are grouped in recoverability level five.

The transactional web services are grouped into five levels and each level exhibits certain kind of recoverability. For example, in level 1, pivot services which cannot be rolled back after their completion. Level 2 holds a service which can be rolled back after their completion. Level 3 is meant for externally interruptible services. The services which are guaranteed to succeed are placed in level 4. Level 5 services are also guaranteed to succeed unless they are cancelled. In each of the levels 3, 4, and 5, two kinds of transactional services with same recovery cost, one which cannot be rolled
back and the other one which can be rolled back on completion have been placed. Level 1 has the lowest recovery cost whereas level 5 corresponds to the most expensive recovery. Thus, the transactional properties are abstracted to the analyst in terms of recoverability levels as shown in Figure 4.1.

![Figure 4.1 Proposed recoverability levels](image)

### 4.3 EMPIRICAL VALIDATION OF RECOVERY COST

In order to measure the recovery costs incurred by different recovery mechanisms, several compositions involving 10, 20, 30, 40, and 50 services have been considered. The services are deployed in Glassfish web server of different machines each with 2.4 GHz Intel Core2 Duo processor and 4 GB RAM, connected in the Intranet. In each composition, a failure of some service \( i \) is considered and its recovery cost is estimated. Since a composite service comprises of component services of different execution times, for a composition of \( n \) services in which \( i^{th} \) service is assumed to fail, the following scenarios have been considered. Basically, the services are of two types viz., normal services (short running) and long running services.

**Scenario 1:** All services upto \((i-1)^{th}\) service are normal and \( i^{th} \) service is also normal
Scenario 2: All services up to \((i-1)^{th}\) service are normal and \(i^{th}\) service is long running

Scenario 3: All services up to \((i-1)^{th}\) service are long running and \(i^{th}\) service is normal

Scenario 4: All services up to \((i-1)^{th}\) service are long running and \(i^{th}\) service is also long running

The recovery time required to restore from the failure of a pivot service is negligible. As described in the previous Section and from Figure 4.1, the recovery time of a service with property \(\{cpcc\}\) is the same as that with property \(\{cc\}\). The recovery time of a service with property \(\{cpr\}\) is the same as that with property \(\{r\}\). The recovery time of a service with property \(\{cpccr\}\) is the same as that with property \(\{crr\}\). Hence, recovery times have been measured for the failure of services with properties \(\{cp\}\), \(\{cc\}\), \(\{r\}\), and \(\{crr\}\). Suitable transactional properties have been assumed for the failed service in order to apply the corresponding recovery mechanisms.

4.3.1  Experiment 1

The recovery time for recovering from the failure of 25\(^{th}\) service in a composition of 30 services, considering four scenarios listed above, are tabulated in Table 4.2. The comparison of recovery time for different recovery mechanisms in four scenarios is graphically depicted in Figure 4.2. It is evident from the graph that the recovery times incurred in Scenario 1 to scenario 4 are in the increasing order. The data available in Table 4.2 are used to compare scenario wise recovery costs for different recovery mechanisms by using the graph shown in Figure 4.3. The compensation time for Scenario 1 and Scenario 2 are almost the same, since both of them involve compensating the same set of normal services. The compensation time for Scenario 3 and Scenario 4 are almost the same, since both of them involve
compensating the same set of long running services. The cancellation time for Scenario 2 and Scenario 3 are more than that in Scenario 1 and Scenario 4 respectively, since cancellation time of a long running service is more than that of a normal service. A similar observation is made with respect to the remaining recovery types as retrying the long running service incurs more time than that of a normal service.

### Table 4.2 Recovery costs of transactional web services

<table>
<thead>
<tr>
<th>Recovery Type</th>
<th>Scenario 1</th>
<th>Scenario 2</th>
<th>Scenario 3</th>
<th>Scenario 4</th>
</tr>
</thead>
<tbody>
<tr>
<td>Backward</td>
<td>2.426</td>
<td>2.567</td>
<td>17.751</td>
<td>17.731</td>
</tr>
<tr>
<td>Cancellation</td>
<td>2.843</td>
<td>3.917</td>
<td>23.901</td>
<td>24.602</td>
</tr>
<tr>
<td>Forward</td>
<td>3.154</td>
<td>4.437</td>
<td>25.776</td>
<td>34.237</td>
</tr>
<tr>
<td>Forward and Cancellation</td>
<td>4.873</td>
<td>6.286</td>
<td>42.567</td>
<td>51.549</td>
</tr>
</tbody>
</table>

![Figure 4.2 Recovery time incurred for different recovery mechanisms](image-url)
4.3.2 Experiment 2

The second testing is performed to compare the recovery costs when the failed or cancelled service is composed in sequence or in parallel. A sequential composition of 30 services where 24 services have completed their execution and 25th service has failed, is considered. Another composition where 20\textsuperscript{th} to 25\textsuperscript{th} services are running in parallel is considered. In this composition, the recovery cost due to the failure of 20\textsuperscript{th} service is considered, since the number of completed services (including concurrent services) is 24 at the time of failure and is comparable to the sequential composition. For both compositions, the average compensation and cancellation costs of the failed service for four different scenarios have been measured. The recovery costs by applying different recovery mechanisms are measured for both compositions in four scenarios and average values are plotted in Figure 4.4. It is observed that the forward recovery cost of the failed service is the same when it is composed in sequence or in parallel. The backward and cancellation recovery cost of the failed service is less expensive when it is
composed in parallel, since the compensation or cancellation logic is applied simultaneously for concurrent services.

![Comparison of recovery costs when the failed/cancelled service is composed in sequence or in parallel](image)

**Figure 4.4** Comparison of recovery costs when the failed/cancelled service is composed in sequence or in parallel

### 4.3.3 Experiment 3

As a third testing, the recovery costs incurred by different recovery mechanisms have been compared. The recovery times of recovering from the failure of \((n-5)th\) service in compositions of \(n\) services where \(n = 10, 20, 30, 40, 50\) are measured 10 times for each of the four different scenarios listed above and the average is considered. The average recovery time for each type of recovery in compositions of \(n\) different services are tabulated in Table 4.3 and plotted as a graph in Figure 4.5. While recovering from the failure of \(i^{th}\) service, it is observed that backward recovery is the least expensive; forward recovery followed by cancellation recovery is the most expensive;
cancellation recovery is more expensive than compensation; forward recovery is more expensive than cancellation.

Table 4.3 Comparison of average recovery costs of transactional web services for different scenarios

<table>
<thead>
<tr>
<th>No. of Services</th>
<th>Failed Service $i$</th>
<th>$RT_{cp}^i$</th>
<th>$RT_{cc}^i$</th>
<th>$RT_r^i$</th>
<th>$RT_{cerr}^i$</th>
</tr>
</thead>
<tbody>
<tr>
<td>10</td>
<td>5</td>
<td>2.146</td>
<td>2.694</td>
<td>3.600</td>
<td>5.554</td>
</tr>
<tr>
<td>30</td>
<td>25</td>
<td>10.269</td>
<td>13.816</td>
<td>17.006</td>
<td>26.319</td>
</tr>
<tr>
<td>40</td>
<td>35</td>
<td>16.563</td>
<td>17.423</td>
<td>24.597</td>
<td>33.484</td>
</tr>
<tr>
<td>50</td>
<td>45</td>
<td>26.015</td>
<td>27.200</td>
<td>32.508</td>
<td>39.339</td>
</tr>
</tbody>
</table>

Figure 4.5 Recovery costs of different recovery mechanisms
From the empirical analysis, it is evident that compensation, cancellation, forward, and forward followed by cancellation recovery techniques incur costs in the ascending order. Hence, their grouping into recoverability levels in the increasing order, as proposed in Section 5 is justified.

4.4 SUMMARY

In this Section, an abstract way of expressing the transactional properties of web services in terms of their recoverability is proposed. The proposed recoverability levels are grouped in the increasing order of service recoverability and recovery costs. The recovery cost for different recovery mechanisms in case of a service failure in compositions of various sizes have been measured and compared with the estimated recovery time.

The proposed approach for transactional requirements abstraction is compared with the related works in Table 4.4. It can be seen from the table that none of the existing works except Benatallah et al’s (2004) work abstracted the transactional behavior of services. Though Benatallah et al’s framework on conversation modeling help developers define extended service models and richer web service abstractions, the meta-model can be analyzed only after the functional requirements are known. The abstractions do not relate to recoverability and recovery cost of services. In contrast, the recoverability level concept proposed in this thesis abstracts the transactional properties of services based on their recovery cost.

The gradation of recoverability levels is clearly justified using the empirical experiments. The recoverability levels help the business analyst to choose a suitable level for the services, based on the expected behaviour of the services and the associated recovery cost. The abstraction of transactional requirements provides the expressiveness to describe the behaviour of services. Since the abstraction is based on the cost of recovery, the business
Table 4.4 Comparison of proposed abstraction for transactional properties with existing work

<table>
<thead>
<tr>
<th>Related Work</th>
<th>Specification of Transactional Requirements</th>
<th>Shortcomings</th>
<th>Proposed Work</th>
</tr>
</thead>
<tbody>
<tr>
<td>WSCL (Banerji et al 2002 WSCI (2002))</td>
<td>As a sequence of service invocations</td>
<td>No relevant abstractions to model service behaviours</td>
<td>Abstractions to specify transactional properties</td>
</tr>
<tr>
<td>Jiuxin et al (2010)</td>
<td>As Conception-constraints</td>
<td>It is difficult for a business analyst to express the business requirements in the form of constraint rules.</td>
<td>Recoverability based abstraction makes specification of requirement easier.</td>
</tr>
<tr>
<td>Chalin et al (2008)</td>
<td>As an extension of use cases</td>
<td>Not suitable for long-running transactions and it does not support specification of transactional properties or dependencies</td>
<td>Models the behavior of normal and long running services.</td>
</tr>
<tr>
<td>Gaaloul et al (2010)</td>
<td>Using transactional dependencies</td>
<td>Specification of all kinds of dependencies between every pair of services is a tough task.</td>
<td>Abstraction is based on recoverability</td>
</tr>
<tr>
<td>Benatallah et al (2004)</td>
<td>Abstractions that build on current standards</td>
<td>1. Inability to specify the abstractions before obtaining functional requirements 2. Not related to recovery of services</td>
<td>1. Transactional requirements can be specified irrespective of functional requirements 2. Abstraction is based on recoverability</td>
</tr>
</tbody>
</table>

analyst is better aware of the recovery cost while specifying the transactional requirements. The abstraction of recoverability levels helps in finding alternate services that are functionally equivalent and behaviourally compatible when the required service is not available. In addition, abstractions are useful to derive a set of required transactional properties of component services which will in turn be used to derive the desired behaviour at the composite service level.