CHAPTER 4

INTEGRATED DESIGN OF WEB SERVICE ACCOUNTING ARCHITECTURE

This chapter presents and explains the integrated design of the accounting modules: metering, billing, accounting and auditing. WS-RADIUS protocol has been designed to meter the web service usage at the service providers and the design details have been explained. The proposed design has been implemented and tested using a sample E-book application. The performance of the sample application has been analyzed and the results have been presented.

4.1 DESIGN OF INTEGRATED ACCOUNTING MODULES

The proposed design of the accounting modules and their integration is given in Figure 4.1.

The metering module is responsible for measuring the web service usage. The metering module meters and store the usage data records in the defined format, and maintains the data on a per service per user basis. The metered data is used for generation of bills to the user and for query and auditing. WS-RADIUS protocol has been developed to meter the web service usage. IPDR standard has been used to design the format and content of usage and accounting data.
Billing is the process of using the accounting information to generate bills on users, for the service provided to them by the service providers. The billing module automates the function of raising bills on the users.

![Figure 4.1 Integrated Design of Accounting Modules](image)

**Figure 4.1 Integrated Design of Accounting Modules**

Accounting policies are used to configure the billing function and define the type of the bill, billing periodicity, content of the bill, payment
date, the method of payment etc. IPDR standard has been used to design the format and content of the bill.

Billing is triggered by the metering module. This trigger is generated based on the charging and accounting policies chosen by the user. For example, if the user has selected the “pay-per-use” option, the tariff requires the user to pay only for the amount of a service really used and the bill would be generated at the end of every usage session.

Bills are prepared using the information from the CAP, UDR and AI databases. The applicable tariff scheme for a user is extracted from the CAP database, and charging is done for the service usage obtained from UDR database. The address and other details for raising a bill are obtained from the AI database. The billing process combines all the relevant customer detailed records in a bill, according to the agreed format and sends the compiled invoice to the appropriate recipients. Different types of billing can be supported as per service provider/user preferences. For instance, billing could be for a single usage or consolidated monthly, bi-monthly, quarterly, half yearly or annual basis. The standard IPDR format has been used to define the bill format.

Payment is automated using the payment gateway process. Payment for services consumed is made by the user to the service provider using the payment gateway facility. The gateway facilitates the transfer of money from the user to the service provider. IPDR standard has been used to design the format and content of the payment data.

The payment mode is used to specify whether the payment is pre-paid or Post-paid. In a pre-paid scenario, the user has to pay money in advance in to their account and can use a service as long as there is sufficient money for the service usage. When the amount paid-in is exhausted the user
would have to make additional payments to continue using the service facility. In a post-paid scenario, the user makes use of the services, a bill is raised for the service usage and the user then makes the payment using the payment gateway.

Accounting is the process of filtering, collecting and aggregating the information of resource usage. The function of the accounting module is to maintain the bills, update the payments and match the bills with the payments to determine the payment outstanding from the users. The accounting module can be used to periodically generate the due/overdue details, along with the payment remainders to the users.

Auditing is the process of verification of the correctness of procedures regarding the service usage, charging and payment. The accounting information available in the database provides valuable information to determine the service usage, verify billing and match the payment received with the bills raised. The auditing module provides an interface to the users for verification of the service usage, charging process and accounting information. The audit module can also be used by third party auditors to query the accounting process and information.

4.1.1 Accounting Record Formats

IPDR (Internet Protocol Detail Record) is a telecommunication standard that has been proposed for exchanging accounting related data between different providers and different accounting systems. IPDR provides a flexible structure to describe the usage attributes, and a common format for record the usage data (Bella et al 2005). The structure of IPDR allows one or many usage records which are service specific and identifies five typical attributes “who, what, where, when and why” values to describe a particular usage event (Zhang 2007).
The IPDR Document hierarchy allows an IPDRDoc to contain a header and multiple IPDR elements (Cotton 2000). Figure 4.2 shows the graphical representation of the different IPDR elements and their relationship.

![Figure 4.2 IPDR Elements](image)

The header contains details such as the docID, version, creation time. The details specific to a service, details about the consumer, service elements, and usage attributes are contained within each IPDR element. The master schema does not define any child elements for the IPDR element because these details are specific to a particular service and in fact may be different for each service. IPDR provides two methods for encoding data, XML (text based) and XDR (compact encoding based) (Andrew 2009).

The metered data has been designed to record data relating to: Who (responsible for the usage); What (service usage measures and quantities); and When (Start Time/End Time). The usage-data is stored in standard XML
form, as XML has the advantage of being human-readable and highly flexible. A sample web service UDR is shown in Figure 4.3.

```xml
<? xml version="1.0"?>
<IPDRDoc
 xmlns="http://www.ipdr.org/namespaces/ipdr"
xmns:xsi="http://www.w3.org/2000/10/XMLSchema-instance"
xsi:schemaLocation="http://www.ipdr.org/namespaces/e-book.xsd"
docId="g8e0ca84-2222-11b2-85ef-fd66246596bb"
Creation Time="2013-11-11T11:10:00Z"
IPDRRecorderInfo ="TEST"
version="3.1">
<IPDR>
<seqNum>1</seqNum>
<IPDR Creation Time>2013-11-11T11:10:00Z</IPDR CreationTime>
<User Name>XYZ</User Name>
<Web Service Provider Name>ABC</Web Service Provider Name>
<Web Service Name>E-book Download</Web Service Name>
<Start Time>2013-11-11T11:10:00Z</Start Time>
<End Time>2013-11-11T11:23:00Z</End Time>
<Usage measures>
<Download Size_MB>14.49</Download Size_MB>
</Usage measures>
</IPDR>
</IPDRDoc>
```

**Figure 4.3 Sample Web Service UDR in IPDR Format**

It can be noted that the sample UDR records the download usage measures. The download size is recorded as 14.49MB for service E-book download. Bill is generated to record the data relating to: Who (user name, billing address); What (service usage measures and quantities and bill amount); and When (Start Time/End Time). A sample bill is shown in Figure 4.4.
Figure 4.4 Sample Bill for E-book Reading Service in IPDR Format
The sample bill has been raised on user XYZ for E-book reading service. This bill shows the amount incurred for the month of November 2013 and has been raised on 3/12/2013. The bill amount is Rs.1000. The bill also shows the outstanding amount due from the user on previous bills raised. Figure 4.5 shows a payment record in IPDR format.

```xml
<IPDRDoc>
docId="g8e0ca84-2222-11b2-85ef-fd66246596bb"
Creation Time="2013-12-10T04:15:03Z"
IPDRRecorderInfo ="PAYMENT"
version="3.1">
<IPDR>
<IPDR Creation Time>2013-12-10T04:15:03Z </IPDR Creation Time>
<seqNum>1</seqNum>
<Username>XYZ</Username>
<UserID>01</UserID>
<Web Service Provider Name >ABC</Web Service Provider Name >
>Total Amount>1000</Total Amount >
<Payment Mode>ECS</Payment Mode >
</IPDR>
</IPDRDoc>
```

Figure 4.5 Sample Payment in IPDR Format

The payment data includes payment amount and payment mode along with details of service provider name and service name for which the payment is made.

4.1.2 WS-RADIUS Protocol

WS-RADIUS protocol has been designed and developed to meter the web service usage of service consumers. WS-RADIUS is based on the AAA standard RADIUS protocol that was designed for authentication and accounting in telecommunication systems. The concepts of client
authentication, message exchanges and message packets outlined in RADIUS protocol have been used in the design of WS-RADIUS protocol.

WS-RADIUS protocol was designed to consist of server and client components. The server component WS-RADIUS protocol is located in the AAAAC server while the client component of the protocol is located in the AAAAC clients. The client component of the protocol is responsible for metering the service usage at the service providers. The web service provider is considered as the AAAAC client. The server component maintains the information of valid service providers and performs the authentication function that validates the service provider. The metered data is stored and maintained in the AAAAC server.

In the Internet or HTTP traffic, accounting is the direct tracking of packets transferred, once the connection is established, whereas in the web services, the related traffic is a little different, and is being handled as SOAP messages. SOAP message’s request and response is the key for metering the web services. SOAP message traffic has been intercepted for handling the metering aspects.

The service provider has to be registered with the AAAAC server to enable the web service usage to be metered. The service users have to also register with the AAAAC server to use the services of the service provider. The services offered by the service provider are maintained in the server along with the charging policies for each service. Before using a service, a user has to select a preferred charging policy for the usage of services. The charging policies opted by the user for each service offered by the service provider would be maintained at the server.

The message packets that have been configured and used for metering are shown in Figure 4.6.
At the beginning of a metering session, the Access-request (1) message packet is sent from the Client to the Server to authenticate the service provider. The packet contains the service provider name, user name and service name. A sample packet is shown in Figure 4.7.

<table>
<thead>
<tr>
<th>Packet-Type=1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet-Dst-Port=1812</td>
</tr>
<tr>
<td>Access-Status-Type = Request</td>
</tr>
<tr>
<td>User -Name = &quot;XYZ&quot;</td>
</tr>
<tr>
<td>Service Provider-Name = &quot;ABC&quot;</td>
</tr>
<tr>
<td>Service -Name = &quot;E-book Reading &quot;</td>
</tr>
</tbody>
</table>

The server verifies the following:

- The service provider is authenticated to verify whether the service provider is valid and registered for accounting the usage data.
The service name is verified to check if the particular service is being offered by the concerned service provider.

The user name is also verified to check if the user has registered with the server for using that particular service offered by the service provider and whether the charging policy opted for the user is available.

If authentication is successful, then an Access-accept (2) message packet is sent from server to client. If authentication is unsuccessful, then an Access-reject (3) is sent by the server to the service provider. Metering of usage would not be carried out if authentication is unsuccessful. Sample packets formats are shown in Figure 4.8.

<table>
<thead>
<tr>
<th>Packet-Type=2</th>
<th>Packet-Type=3</th>
</tr>
</thead>
<tbody>
<tr>
<td>Packet-Dst-Port=1812</td>
<td>Packet-Dst-Port=1812</td>
</tr>
<tr>
<td>User -Name = &quot;XYZ&quot;</td>
<td>User -Name = &quot;XYZ&quot;</td>
</tr>
</tbody>
</table>

(a) Access-accept Message Packet (b) Access-reject Message Packet

Figure 4.8  Message Packets Format for Access – accept and Access-reject

To begin the metering process, Start-Accounting-request (4) packet is sent from the client to the server. Start-Accounting- response is sent to the client to acknowledge the start of the metering process. The start time of metering is recorded at the server for the service provider for a particular service and user. At the end of the usage session, Stop-Accounting-request (5) packet is sent from the client to the server. Stop-Accounting- response is sent to the client to acknowledge that the metering process is stopped and the end time of metering is recorded at the server. The message packets format is shown in Figure 4.9.
Packet-Type=4
Packet-Dst-Port=1813
Acct-Session-Id = "4D348AC-0000098"
Acct-Status-Type = Start
Acct-Authentic = RADIUS
User -Name = "XYZ"
Service Provider-Name = "ABC"
Service -Name = "E-book Reading"

Packet-Type=5
Packet-Dst-Port=1813
Acct-Session-Id = "4D348AC-0000098"
Acct-Status-Type = Stop
Acct-Authentic = RADIUS
User -Name = "XYZ"
Service Provider-Name = "ABC"
Service -Name = "E-book Reading"

(a) Message Packets Format for Start-Accounting
(b) Message Packets Format for Stop-Accounting

Figure 4.9 Message Packets Format for Accounting

WS-RADIUS has been designed to meter both atomic and composite web services. In a composite web service, the individual services could be offered by one service provider or by multiple service providers.

4.1.3 Billing

Billing is the function that transforms the charging information into invoices for customers. A charging scheme is an instruction for calculating a charge. Usually, a charging scheme is represented by a formula that consists of charging variables (e.g. volume, time) and charging coefficients (e.g. price per time unit).

The users are charged based on the policies that they have selected and agreed with the service provider. Different pricing schemes can be chosen by the user, such as, Pay-per-use, content based, time based and function based payment option for web service usage.
Examples have been used to illustrate the payment calculation. Let MC be the Membership charges and FC be the fixed reading charges levied. Let SC be the charge for every one hour of session time, and CC be the charge for reading a book and UC be the charge for function usage.

**Considering a ‘Fixed’ Charging scheme or policy, the payment is calculated as :**

\[
\text{Cost} = \text{MC} + \text{FC}
\]

The cost includes membership fee and fixed amount charged for the usage of services. FC would vary depending on the pricing option that is decided by the user. For example, if a user registers under “group” and selects the option of downloading books with the fixed usage charges of Rs.70/20MB, and if the user downloaded 1 GB, given a membership charge of Rs.70/month for a user, then the charge would be:

\[
\text{Cost / month} = \text{MC} + \text{FC} \\
= 70 + 70 \times 1024 / 20 \\
= \text{Rs.3654}
\]

For example, if the user has used the service for 30 hours, and this option has only a session charge of Rs.15/hour, given a membership charge of Rs.100/month, then the charge would be:

\[
\text{Cost / month} = \text{MC} + \text{SC} \times \text{No. of hours} \\
= 100 + 15 \times 30 \\
= \text{Rs.550}
\]
Considering a ‘Session’ Charging scheme or policy, the payment is calculated as:

\[ SC = \sum SC_i \times Hi \]

The cost includes charges for session and the number of hours of service usage. SC would vary depending on the pricing option that is decided by the user. For example, if the user has used the read access for 30 h with the usage charges of Rs. 0.50/min, and given a membership charge of Rs.100/month, then the charge would be:

\[
\begin{align*}
\text{Cost / month} &= MC + \sum SC_i \times Hi \\
&= 100 + 0.50 \times 30 \times 60 \\
&= \text{Rs.1000}
\end{align*}
\]

Considering a ‘Volume’ Charging scheme or policy, the payment is calculated as:

\[ CC = \sum CC \times N \]

The cost includes charges for reading a book and the number of books read. CC would vary depending on the pricing option that is decided by the user. For example, If a user has selected the option of accessing special books, with the usage charges of Rs.30 / book and the bi-monthly membership charge of Rs.175, and if the user read 25 special books, then the charge would be:

\[
\begin{align*}
\text{Cost / month} &= MC + CC \times N \\
&= 175 + 30 \times 25 \\
&= \text{Rs.925}
\end{align*}
\]
Considering the ‘Function’ Charging scheme or policy, the payment is calculated as:

\[ UC = \sum F_{Ni} \]

The cost includes the charge for function. The charges would be different for different functions. For example, the charge for a download function would be different from the rate charged for reading an archived book. The function charge is the sum total of charges for all the functions used. For example, if a user has selected the option of read and download the books from archival, with the usage charges of Rs.600/ for 1024MB and Rs.25/ for an additional 20 MB and if the user has downloaded 1044 MB, then the charge would be:

\[
\text{Cost / month} = F_{Ni} \\
= 600+25\times\frac{20}{20} \\
= \text{Rs.625}
\]

4.2 IMPLEMENTATION AND PERFORMANCE ANALYSIS

A sample E-book application was implemented to test the proposed architecture and design of the accounting modules. Web services were used to implement the application. Both atomic and composite web services were used in the sample application. The description of the web services are given in Table 4.1.
### Table 4.1 List of Web Services for E-Book reading service Application

<table>
<thead>
<tr>
<th>S.No.</th>
<th>Web Services</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Search function</td>
<td>Allows the members to search for specific information or books</td>
</tr>
<tr>
<td>2.</td>
<td>Read access</td>
<td>Allows members to read any or all of the books displayed</td>
</tr>
<tr>
<td>3.</td>
<td>Download access</td>
<td>Allows members to download an entire book or a few chapters or pages</td>
</tr>
<tr>
<td>4.</td>
<td>Read/download the books from archival</td>
<td>Read/download the books from archival service are a composite web service, comprising of the reading and downloading. Members to read/download the books that are archived</td>
</tr>
<tr>
<td>5.</td>
<td>Rating</td>
<td>Allows members to rate a book</td>
</tr>
<tr>
<td>6.</td>
<td>Comment/review</td>
<td>Allows members to comment/review the books they have read</td>
</tr>
<tr>
<td>7.</td>
<td>Sharing</td>
<td>Members can form groups and exchange opinions and track book reviews</td>
</tr>
<tr>
<td>8.</td>
<td>Pre-order</td>
<td>Pre-order a composite web service, Members are allowed to pre-order a book.</td>
</tr>
</tbody>
</table>

Policies were developed to support different pricing schemes. The charging was based on session time, volume and on functions used and the schemes provided are shown in Table 4.2.
### Table 4.2 List of pricing schemes for E-Book Reading Service Application

<table>
<thead>
<tr>
<th>Charging Schemes</th>
<th>Descriptions</th>
</tr>
</thead>
<tbody>
<tr>
<td>Session based charging</td>
<td>• Fixed - Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Package 3- Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Personal - Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Group - Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Package1&amp;2- Limited access</td>
</tr>
<tr>
<td></td>
<td>• Pay-per-use</td>
</tr>
<tr>
<td>Volume based charging</td>
<td>• Package 2 &amp; 3- Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Group - Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Personal - Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Pay-per-use</td>
</tr>
<tr>
<td>Function based charging</td>
<td>• Package 2 &amp; 3- Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Personal - Unlimited access</td>
</tr>
<tr>
<td></td>
<td>• Pay-per-use</td>
</tr>
</tbody>
</table>

Different payments options have been included in the application. In the pre-paid option, the users are required to pay a deposit amount in advance. The deposit amount is adjusted against web service usage. When the deposit amount is used up, then user would have to deposit additional amounts for the continuing use of the web services. An alert message will be sent to the user. In the post-paid option, the billing for services usage for members could be periodic with option for monthly, bi-monthly, quarterly and half-yearly or on annual basis.

The proposed metering framework was implemented as shown in Figure 4.10.
SERVICE PROVIDER

Figure 4.10 Block Diagram of the WS-RADIUS Protocol for Web Service Metering and Accounting

AAAAC server was configured using Linux operating system Ubuntu 10.04 LTS and the database was configured using MySQL version 5.1.41. The web application was created using open source Java Glassfish server version 4.1 and JAX-WS was used to develop the web services. Apache web server version 2.2.14 was used to host the application at AAAAC client located at service providers.

Free radius-server-2.2.0 was used as the base for the development of WS-RADIUS protocol for implementing metering of web service resource usages. Figure 4.11 shows the snapshot the protocol that displays a ready to process request message.
The next step of client authentication is shown in Figure 4.12 and Figure 4.13. Figure 4.12 shows the snapshot of unsuccessful client authentication. If wrong credentials are presented by the client, then authentication is unsuccessful and an Access-Reject message is sent back to the client.

Figure 4.12 Snapshot of Access-Reject Message in WS-RADIUS Protocol

Figure 4.13 shows the snapshot of successful authentication of client by the server. The snapshot shows that the request sent by client has been accepted and the server is ready to process the requests.

Figure 4.13 Snapshot of Access-accept Message
The snapshot of Session Start is shown in Figure 4.14. It shows that the connection is established and metering would also commence.

```
Packet-Type=4
Packet-Dst-Port=1813
Acct-Session-Id = "4D348AC-00000098"
Acct-Status-Type = Start
Acct-Authentic = WS-RADIUS
User-Name = "XYZ"
NAS-Port = 0
Called-Station-Id = "00-02-6F-AA-AA-AA:My Wireless"
Calling-Station-Id = "00-1C-B3-AA-AA-AA"
NAS-Port-Type = Wireless-802.11
Connect_Info = "CONNECT 48Mbps 802.11b"
```

**Figure 4.14 Snapshot of Message Packets for Session Start**

Figure 4.15 shows the snapshot of Session Stop. The session time, input packets and output packet details are displayed.

```
Packet-Type=4
Packet-Dst-Port=1813
Acct-Session-Id = "4D348AC-00000098"
Acct-Status-Type = Stop
Acct-Authentic = WS-RADIUS
User-Name = "XYZ"
NAS-Port = 0
Called-Station-Id = "00-02-6F-AA-AA-AA:My Wireless"
Calling-Station-Id = "00-1C-B3-AA-AA-AA"
NAS-Port-Type = Wireless-802.11
Connect_Info = "CONNECT 48Mbps 802.11b"
Acct-Session-Time = 30
Acct-Input-Packets = 25
Acct-Output-Packets = 7
Acct-Input-Octets = 3407
Acct-Output-Octets = 867
Acct-Terminate-Cause = User-Request
```

**Figure 4.15 Snapshot of Message Packets for Session Stop**
A sample screenshot of the service usage records is shown in Figure 4.16.

Figure 4.16 Snapshot of a Web Service Usage Record

The usage record entries indicate the web service name, user name, session start time, end time, total session time and total bytes transfered.

A sample screenshot of bill generated in IPDR format is given in Figure 4.17.

Figure 4.17 Sample Snapshot of Bill Generated in IPDR Format
The sample snapshot shows the bill raised on user XYZ for the month of November 2013. The total bill amount computed is also shown in the figure as Rs.1000. Sample screenshot of payment generated in IPDR format is given in Figure 4.18.

Figure 4.18 Snapshot of Payment Record in IPDR Format

Figure 4.19 Snapshot of Accounting Query by Service Provider
The payment record contains the details such as the user ID, service provider name, and payment amount and payment mode. Figure 4.19 shows the snapshot an accounting query given by service provider and the output received.

The query provides the accounting information details such as the user name, user-ID, service provider name, billing mode, total bill amount, bill date and dues outstanding.

4.2.1 Performance Analysis

The performance of the proposed model was analyzed using the sample application. One of the parameters measured was web service response time. The response time was calculated as the time taken from initiation of web service to obtaining the result of execution of web service.

The response time for metering the web services by the WS-RADIUS protocol is given in Figure 4.20. The graphs have been plotted against the number of users and the response time. The maximum number of users that utilized the service was considered to be 10000.

![Response Time](image)

**Figure 4.20** Response Time of Metering Web Services using Meter and Without Meter
From the Figure 4.20 it can be noted that the response time increases when the number of users increases. It can be seen from the graph that the response time of metering using WS-RADIUS protocol is more when compared with the metering without using WS-RADIUS protocol.

Load UI tool was used to measure the performance of a Web service. Load testing refers to the practice of modeling the expected usage of a web service by simulating multiple users accessing the service concurrently. The snapshot in Figure 4.21 shows the performance of WS-RADIUS protocol, where the X axis refers to number of users and Y axis refers to the number of requests.

![Figure 4.21 Load Test Run on the WS-RADIUS Web Services](chart)

Violet color represents the average response time, yellow represents the requests, and green shows the throughput per second. The performance has been compared for request/response count and time in seconds. It can be noted, that the average response time and throughput remains more or less constant even when the number of requests increases.

The revenue for session based charging is given in Figure 4.22.
Figure 4.22 Revenue Based on Session Based Charging

It can be seen from the graph that the flat rate of charging yields a fixed revenue to the service provider. However, the user would need to be charged based on their actual usage. When the usage is heavy, as shown as 5 hrs./day, then the revenue earned, if the user is charged on a per hour basis is more than that of the flat rate. Hence the service provider gains more revenue, when the usage is less as shown as 1 hour/day, the user need not pay for hours not used by them.

The revenue for volume based charging is given in Figure 4.23.
From the Figure 4.23, it can be noted that in volume based charging also, the service provider gains when the usage is heavy, while the user only pays for their usage.

The revenue earned for sample function based charging is given in Figure 4.24.

![Function Based Charging](image)

**Figure 4.24 Revenue Based on Function Based Charging**

From the Figure 4.24, it can be noted that in function based charging also, the service provider gains when the usage is heavy, while the user only pays for their functional usage.

### 4.3 CONCLUSION

The integrated design of the accounting modules: metering, billing, accounting and auditing are discussed in detail in this chapter. The design details of WS-RADIUS protocol used in metering web service usage has been presented. A sample application was developed and tested to analyze the performance of the proposed design. The implementation of WS-RADIUS protocol and the accounting modules has been explained. The impact on revenue when the user is charged based on the usage has been illustrated using graphs for the sample application.