QANTAS

<Use Case Realization: DataServices>

Version 1.0

REVISION LIST

Ver.	Date Released	Modifications	Author(s) and Department	Reviewer(s) and Department
1.0		Initial version	Senthil	

RELATED DOCUMENTS

Document Title	File Name and Location	Summary

TABLE OF CONTENTS

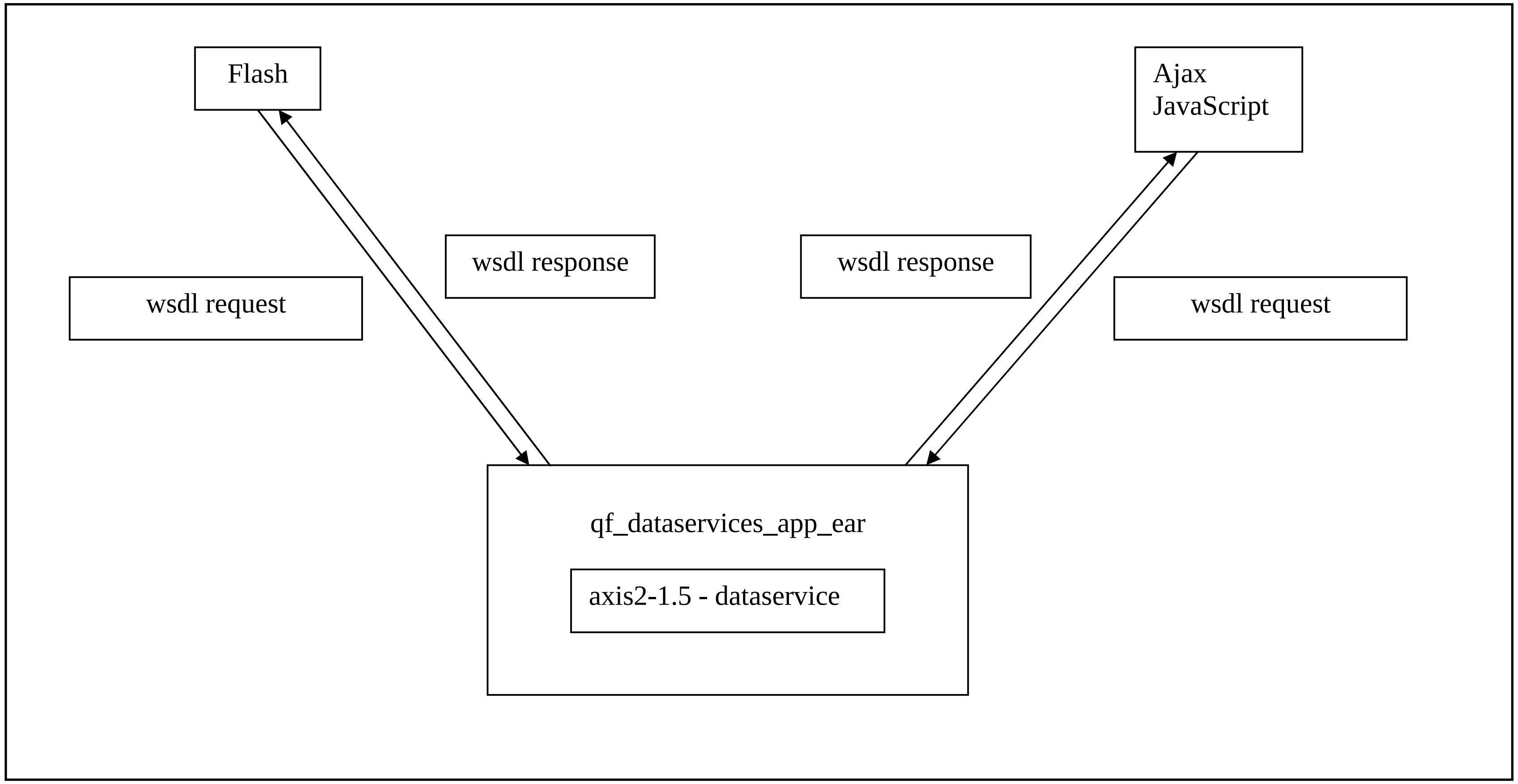
1. Introduction

This document describes the use case realization of dataservices application.

2. Requirements

Currently Data in Qantas.com is available through diverse applications. The Data should be aggregated and accessed through a single common access point.

3. Design Elements



3.1 Web Service

Webservice is used to aggregate data. Axis version axis2-1.5 is used. qf_dataservices_app_ear application hosts the webservice - dataservice.

3.2 Wsdl

Wsdl file used to define the dataservice.

dataservice.wsdl

3.3 Xsd

Xsds used in the Wsdl file.

RouteMapsDataService.xsd

PricingDataService.xsd

xsds are referenced in the wsdl file through importing respective xsd schemas. This approach will help in plugging the source data xsds into the main wsdl without any major modifications in the wsdl.

3.4 Services.xml

services.xml created with the help of axis2-1.5 class generators. This will define the service.

services.xml

3.5 Axis2-1.5

axis2-1.5 webservice engine is used for the deployment. This version used is compatible with the weblogic server 10.1 and java jdk - 1.5

3.6 Implementation classes

For every application, a separate implementation class is introduced. In future also, it is recommended to introduce a separate implementation class, whenever a new application has to be included in the web service.

Pricing
- Service interface – PricingDataService
- Service implementation – PricingDataServiceImpl

Routemaps
- Service interface – RouteMapsDataService
- Service implementation – RouteMapsDataServiceImpl

3.7 Pricing data

Sample URL for the pricing data, which the application will hit and get the result, is

http://www.qantas.com.au/deals/do/dyn/specials/querycurrentdeals.xml?quantity=1&departureAirportCodes=SYD&arrivalAirportCodes=PER&cheapestDealPerDistinctAirportPair=true&QF_Compression=false&travelDate=20091029&marketingRegionCode=AU&dealTypeCode=R

These two parameters values have been fixed in the application
cheapestDealPerDistinctAirportPair=true
QF_Compression=false

3.8 RouteMaps data

Sample URL for the RouteMaps data, which the application will hit and get the result, is

http://www.qantas.com.au/flash/routemaps08/int/citiesGuide_int_en_AU.xml

3.9 JS and Flash

Webservice clients will be written at the Flash and javascript end at the initial stage to access the service. The results returned will be then passed through xslts to get the desired values.

In future the client access could be extended to many parties, without any modifications in the service. Anyhow, on doing so, the performance capability of the service and the environment should be taken to consideration.

3.10 dataservices.aar

dataservices.aar consists of the following components

- ➔ java classes for data service
  - o java classes created using axis2-1.5

    java classes have been created using the following wsdl2java command in axis2-1.5/bin folder

    ```
    wsdl2java -t -ss -sd -g -u -ssi -uri dataservice.wsdl
    ```

    this requires both the wsdl and corresponding xsds should be available in the axis2-1.5/bin folder.

    this will generate both the server side and client side classes. Care should be taken while copying the classes to clearcase, since the skeleton and skeleton interface are modified to accommodate the business logic of the service.

  - o implementation classes for the service
    - ▪ each application data has been accessed via a separate implementation class. In future, when we introduce more application data, it is recommended to introduce separate implementation class for respective applications.

  - o dataservice utility class

        - this utility class is having static methods for creating the url patterns, accessing the http urls and get the result.

- ➔ META-INF folder
    - META-INF folder in turn consists of the following components
        - INDEX.LIST
        - MANIFEST.MF
        - dataservice.wsdl
        - services.xml
        - PricingDataService.xsd
        - RouteMapsDataService.xsd

3.11 Application structure

The application has been structured as an enterprise application, which in future can accommodate any enterprise classes, if required.

qf_dataservices_app_ear

ear consists of

META-INF

application.xml

weblogic-application.xml

APP-INF

lib – consists of axis2-1.5 libraries and application dependent libraries

During startup application requires certain axis libraries to make the application active. Hence the libraries are placed at app-inf/lib instead of web-inf/lib. When we place the libraries at web-inf/lib, the application couldn’t able to refer the libraries before the application gets activated.

classes – consists of properties files

This has been done, since emove/pmove scripts wont allow content deployment of properties files when it is placed in web-inf/classes. Once if we can able to move contents such as xml,xsd,properties files to web-inf/classes, it can be moved there.

The properties file currently deployed provides the information about the respective domain host.

qf_dataservices_app_war

```
resources
    jsp
      - application specific jsps will go here
WEB-INF
    conf
        axis2.xml
    modules
        modules.list
        addressing-1.5.mar
        axis2-jaxws-mar-1.5.mar
        axis2-scripting-1.5.mar
        mex-1.5.mar
        mtompolicy-1.5.mar
        ping-1.5.mar
        soapmonitor-1.5.mar
    services
        services.list
        dataservices.aar
        version-1.5.aar
    beehive-netui-config.xml
    web.xml
    weblogic.xml
```

3.12 Web Server Configuration

Webserver rule for ***/data/*** is added in obj.conf to route the dataservices application requests to the respective domains.

3.13 Akamai Cache Configuration

To be filled once configuration is done.

4. Known Problems and Troubleshooting

- ➔ If the webservice is returning exception in fault string, it might be due to wrong parameters passed which forms the basis for the formation of URL. Initial step is to form the application specific http URL with the parameters passed via webservice and to check the result by hitting the URL directly.

5. Future enhancements

- ➔ The idea for this webservice implementation is to serve all the data provided by Qantas.com applications. In future it should be expanded to include all the application data and should be the single entry point to get data from Qantas.com.

  XML data which are in consideration are provided in the attached xls.

  xml_list.xls

- ➔ Things to consider when introducing new data in the dataservice
  - Whether the existing data cached in akamai
  - What are possibilities of data in sync when accessed via URL vs when accessed via web service
  - No of hits expected to the webserver/weblogic server should be analysed.
  - Response time of the result data should be analysed.