How to Handle Errors in SAP xMII Composite Applications

Applies to:

SAP xMII version 11.5

Summary

This document explains how to handle errors in Business Logic Transaction in SAP xMII. It gives a detailed description and step-by-step guide on how the error can be traced in the BLS transaction and how it can be logged and displayed to the user.

Author(s): Dipankar Saha

Company: IBM India Pvt. Ltd

Created on: 06 November 2006

Author Bio

Dipankar Saha is currently working in IBM India as Advisory System Analyst developing composite applications using SAP xMII. Previously he worked for SAP Labs India and was involved in development of xMII product and ISA95 business package. Dipankar has about 5 years of experience in application software development.

Table of Contents

Introduction

Composite applications for plant floor visualizations and factory data collection can be developed easily using SAP xMII. To develop the SAP xMII composite applications the business logic needs to be developed using the SAP xMII Business Logic Service. While developing business logic transactions in SAP xMII developers need to do error handling to log the error messages properly and inform the user about the problem in error situations. To do this no standard method is available in SAP xMII, however, developers can follow the simple procedure as described below to incorporate a error handling logic in Business Logic Transactions in SAP xMII.

Scenario

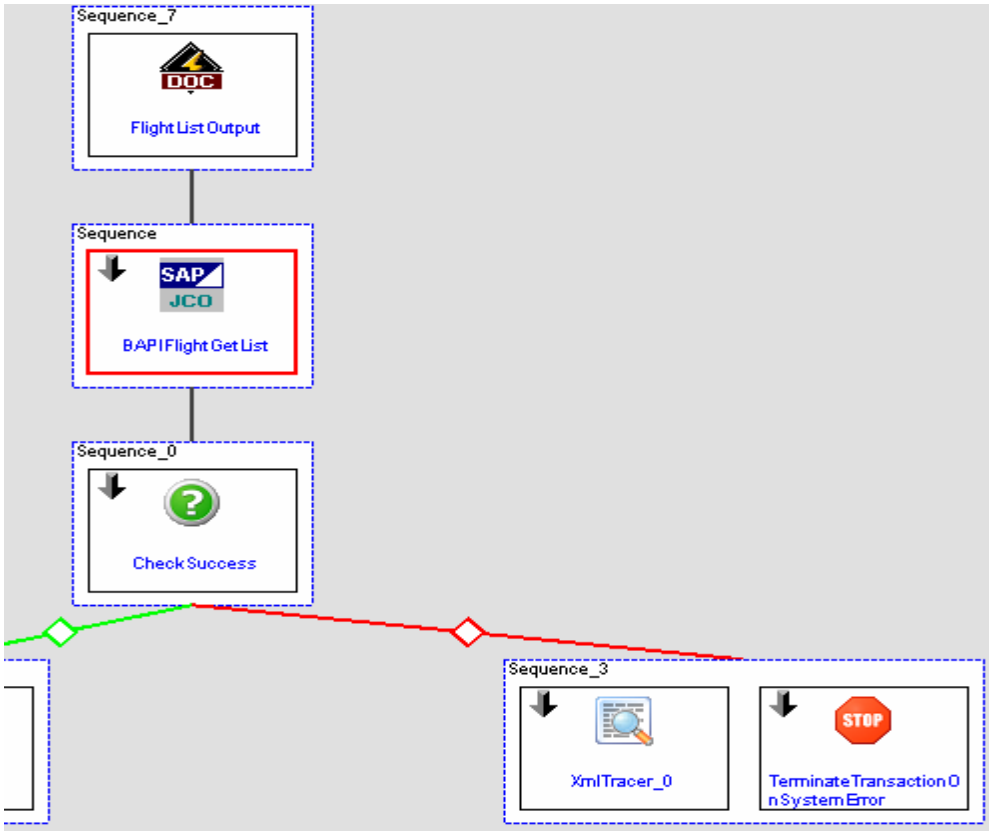
In our example we have a Business Logic Transaction where a SAP BAPI from a backend R/3 system is called using the SAP JCO action block. The RFC returns some data in usual scenario which is processed in the Business Logic Transaction and displayed to the user in a web page. To do this we are using a Xacute Query which calls the Business Logic Transaction and a display template (of type iGrid) where the data is displayed in the web page.

In unusual scenario if the SAP R/3 system is not available (due to network problem) or the RFC returns an error message (due to data error) our aim is to log the message in an application log and inform the user about the problem.

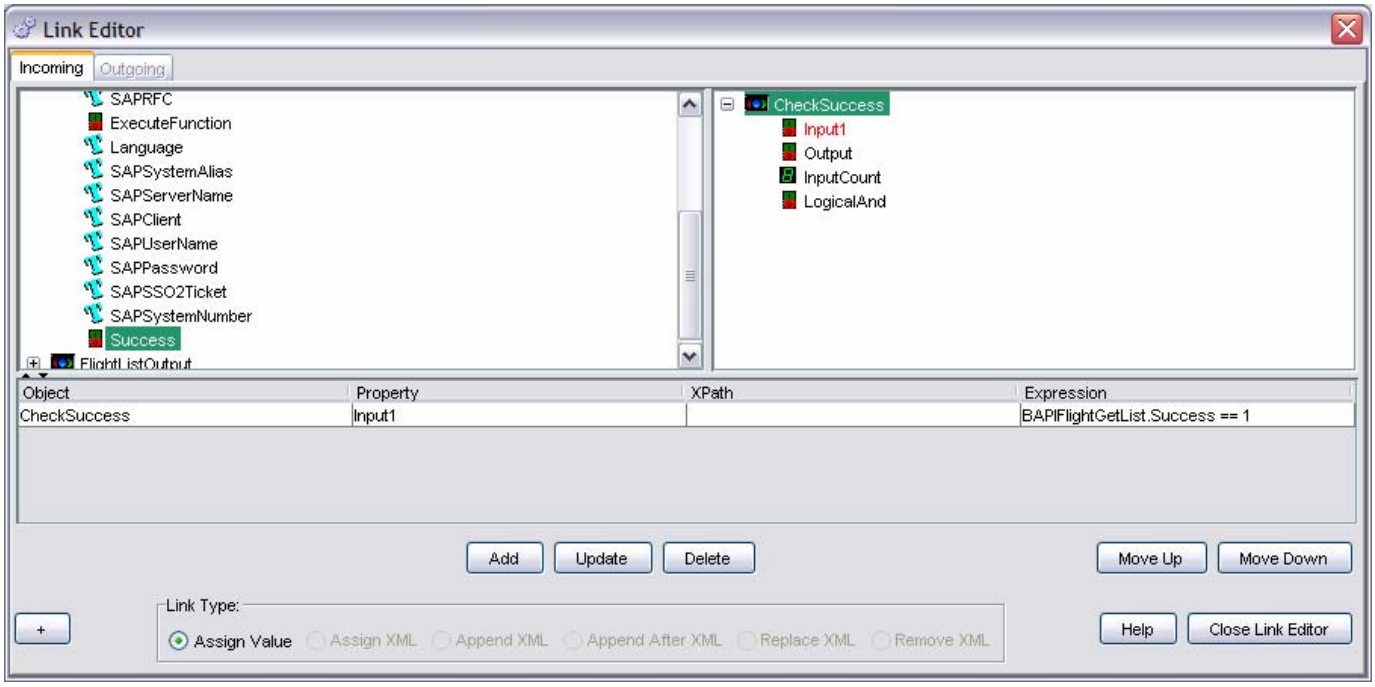
Error Handling

Error handling in Business Logic Transaction

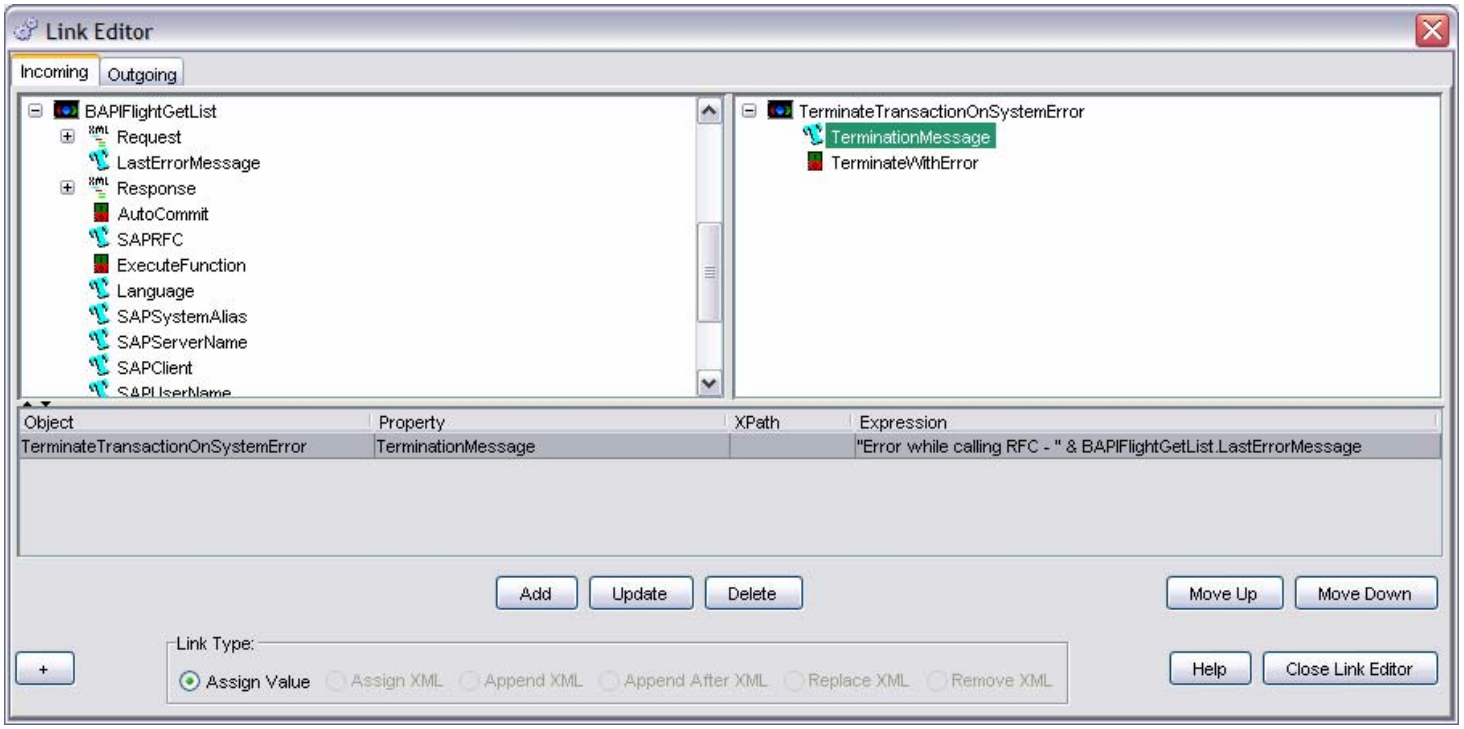
In BLS transaction we check the “Success” flag of the JCO action block (which returns 1 if the RFC call is successful) by the Conditional action block and also the BAPI Return table in the response message.
If the “Success” Flag doesn’t return 1 or if the Return table has an error message then we terminate the transaction by the Terminate Transaction action block and adding an error message.



Terminate on RFC Error

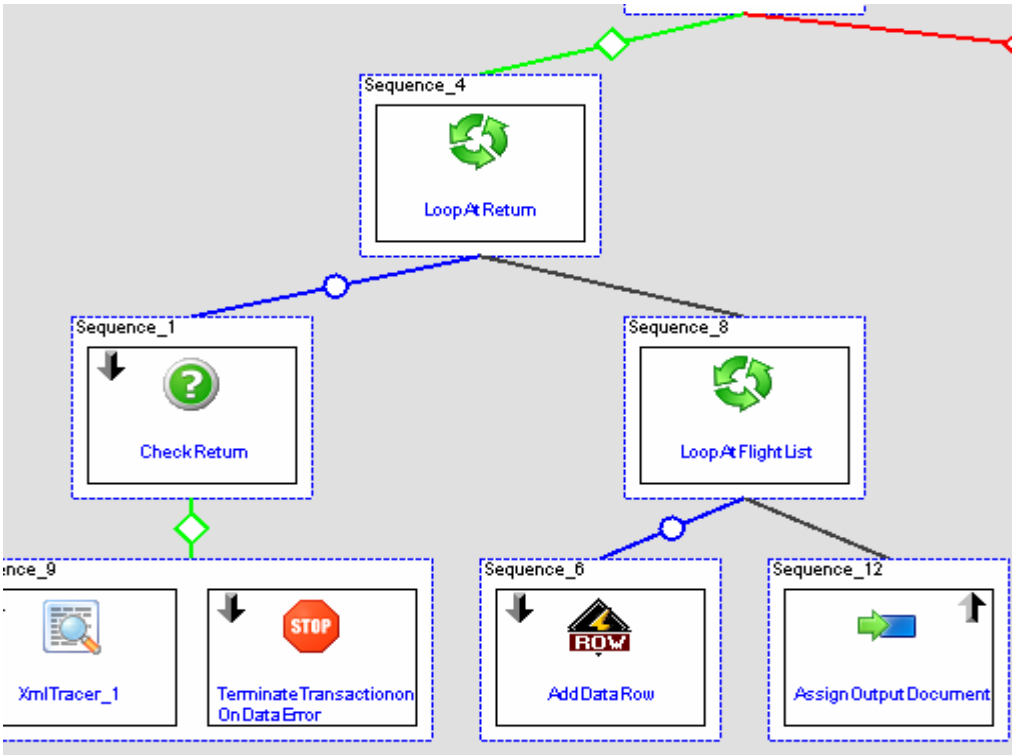


Conditional Action CheckSuccess

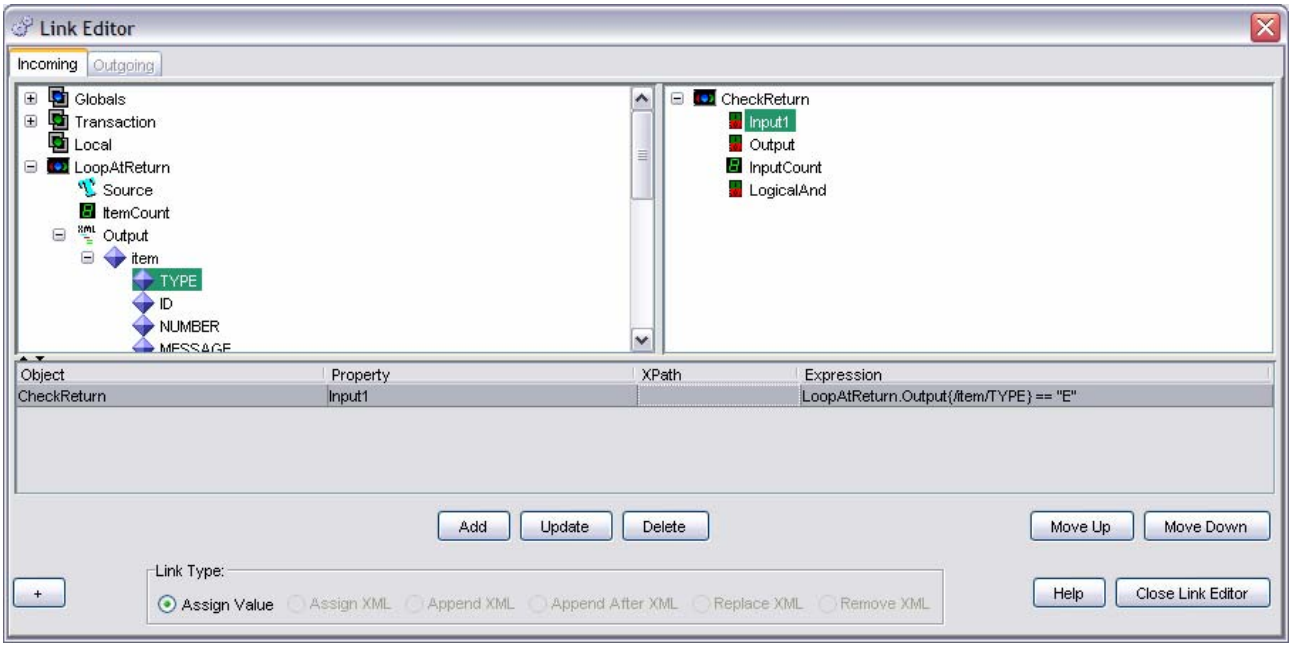


Termination Message

If the JCO action 'Success" flag value is 1, we then proceed to the next step to check any data error. For that we use a Repeater action to loop on the Return table of the BAPI response and check the message type. If the Type is "E" then that means the BAPI has returned an error message due to an application/data error. So we terminate the transaction by logging the error message as below:



Terminate On Data Error



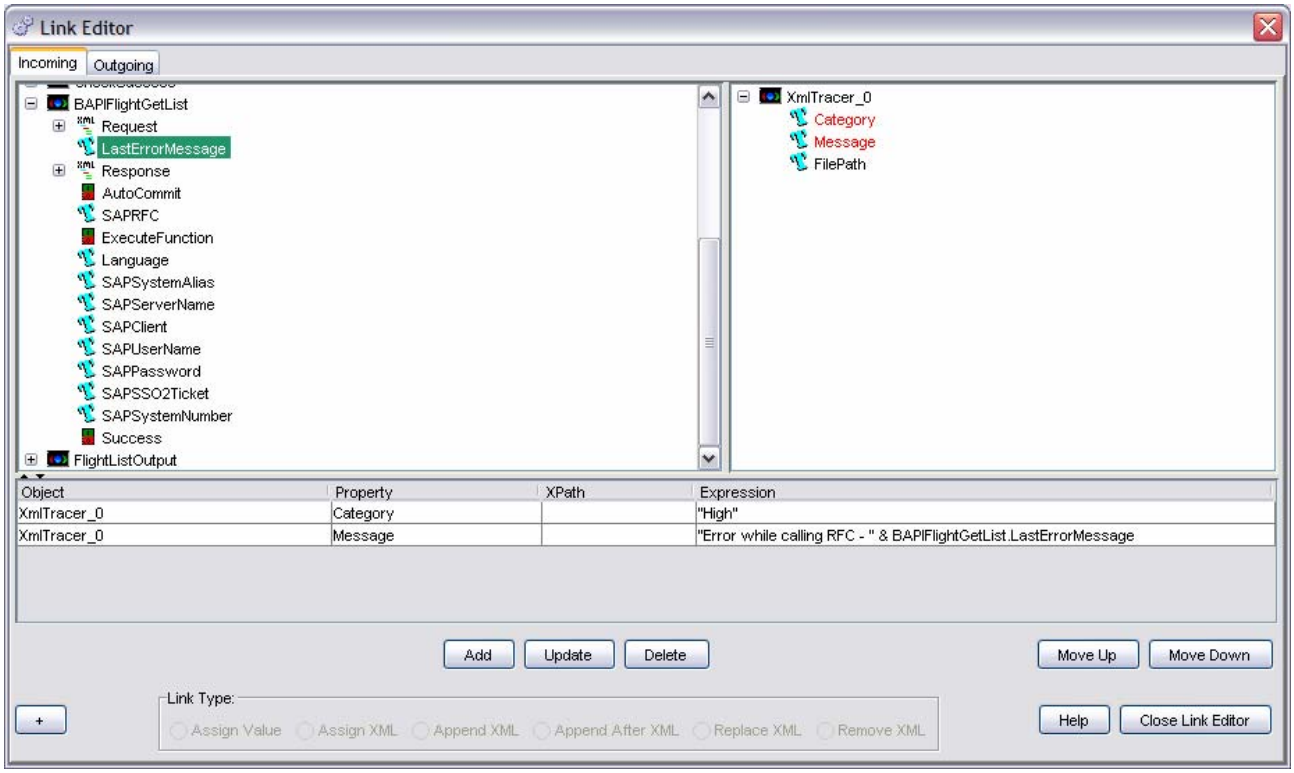
Check Message Type in Return Table

We proceed with the normal processing if no system error or data error is encountered.

We also log the error message in an application log using the XML Tracer action. XML Tracer action saves the log message in a XML file saved in the xMII server which will be displayed using a display template

(iGrid) in the log display screen. To configure the XML Tracer action we need to provide the path in the xMII server where the file will be saved, the message category (which we are using here as log severity) and the actual log message. So whenever the XML tracer action is executed it adds the log message to the specified XML file.

XML Tracer Configuration



XML Tracer Link Editor

Error Handling in irpt Page

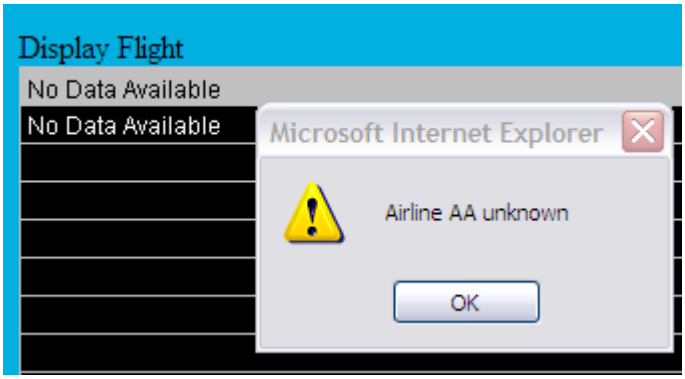
Now we have created a Xacute Query and an iGrid Display Template which we have used in the web page (irpt) to display the output. But as the query object or iGrid/iChart applet doesn't have any method to read the error message (which comes under the <FatalError> node in the query output XML) or doesn't expose the output XML object we need to use an iCommand applet to get the error message. The irpt page source should be as below:

```
//DisplayFlight.irpt
<html>
<head></head>
<SCRIPT LANGUAGE="JAVASCRIPT">

function initialize()
{
   //execute the query
   document.iCommandFlightList.executeCommand();
   //get the error message if any
   var message = document.iCommandFlightList.getLastError();
   //display error
   if(message != "")
   {
     alert(message);
   }
        else  {
       var qTemplate = document.iCommandFlightList.getQueryTemplate();
       document.iGridFlightList.setQueryTemplate(qTemplate);
       alert(document.iGridFlightList.getLastError());
        }
}
</SCRIPT>
<body onload="initialize()">
Display Flight<br>
<APPLET NAME="iGridFlightList" WIDTH="640" HEIGHT="400" CODE="iGrid"
CODEBASE="/Illuminator/Classes" ARCHIVE="illum8.zip" MAYSCRIPT>
<PARAM NAME="DisplayTemplate" VALUE="UserTemplates/PoC/Dipankar/ShowFlightList">
</APPLET>
<APPLET NAME="iCommandFlightList" WIDTH="1" HEIGHT="1" CODE="iCommand"
CODEBASE="/Illuminator/Classes" ARCHIVE="illum8.zip" MAYSCRIPT>
<PARAM NAME="QueryTemplate" VALUE="UserTemplates/PoC/Dipankar/XAFlightList">
</APPLET>
</body>
</html>
```


Note: queryobject does have a method getStatusMessage(), but it doesn't provide the actual error message present in the query output, instead it gives a standard message "No Data Available" in case of any application error other than authorization error.

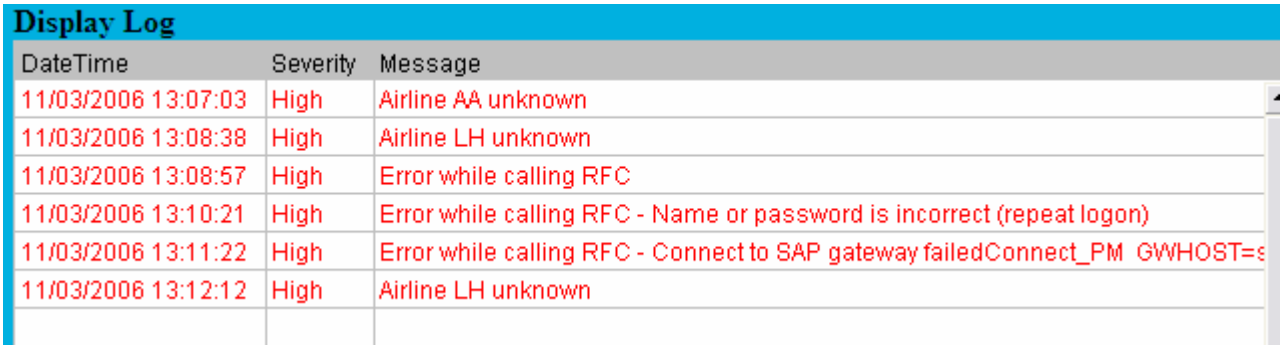
iCommand object provides a method getLastError() which reads the error message present in the query XML output. So first we execute the query by the iCommand object as `iCommandFlightList.executeCommand()` and then check whether any error is returned by the query. If we find an error string returned by `iCommandFlightList.getLastError()` then we display the error message to the user by a Javascript alert as below. Otherwise if no error is found we assign the query template to the iGrid (display template) object to display the data in the web page.



Displaying the Application Log

The application log is written in a XML file using the XML Tracer action in BLS and saved in the xMII server. We display that in a separate screen.

To get the XML log data we have used a XML query (by providing the HTTP URL for the XML file written in the xMII server e.g. http://<servername>/logfolder/flightlog.xml) and a display template of type iGrid to display the data in tabular format as below.



Related Content

SAP XMII Getting Started Guide

SAP xMII Help

How to Create a JCO BAPI RFC Itemset to xMII Rowset Conversion Utility

Disclaimer and Liability Notice

