Hitachi HiCommand® Backup Services Manager

XML Reporting Framework Setup and Customization Guide

November 2006

Version 6.0

Doc ID: MK-95APT011-03

1. Introduction

The Hitachi HiCommand® Backup Services Manager XML Reporting Framework is a powerful complement to the reporting capabilities of the web based GUI. The framework allows you to create custom report specifications which run queries against the Hitachi HiCommand® Backup Services Manager object model. The custom report specifications:

- Can be run at pre-configured times
- Generate reports in text, html, delimited and PDF[1] output formats. Reports can contain text and/or charts
- Allow you to save reports as files or have them emailed to one or more recipients

This document provides an overview of the XML Reporting Framework and comprehensive reference sections to help the Hitachi HiCommand® Backup Services Manager Administrators develop customized reports. This guide is valid for HBSM Portal version 6.0 onward.

Separate instructions are provided as necessary for Windows and UNIX.

If at any time during setup you encounter a problem, or if you simply want to verify a step or have a question about the Reporting Framework please contact the HDS Technical Support Center which will be happy to assist you:

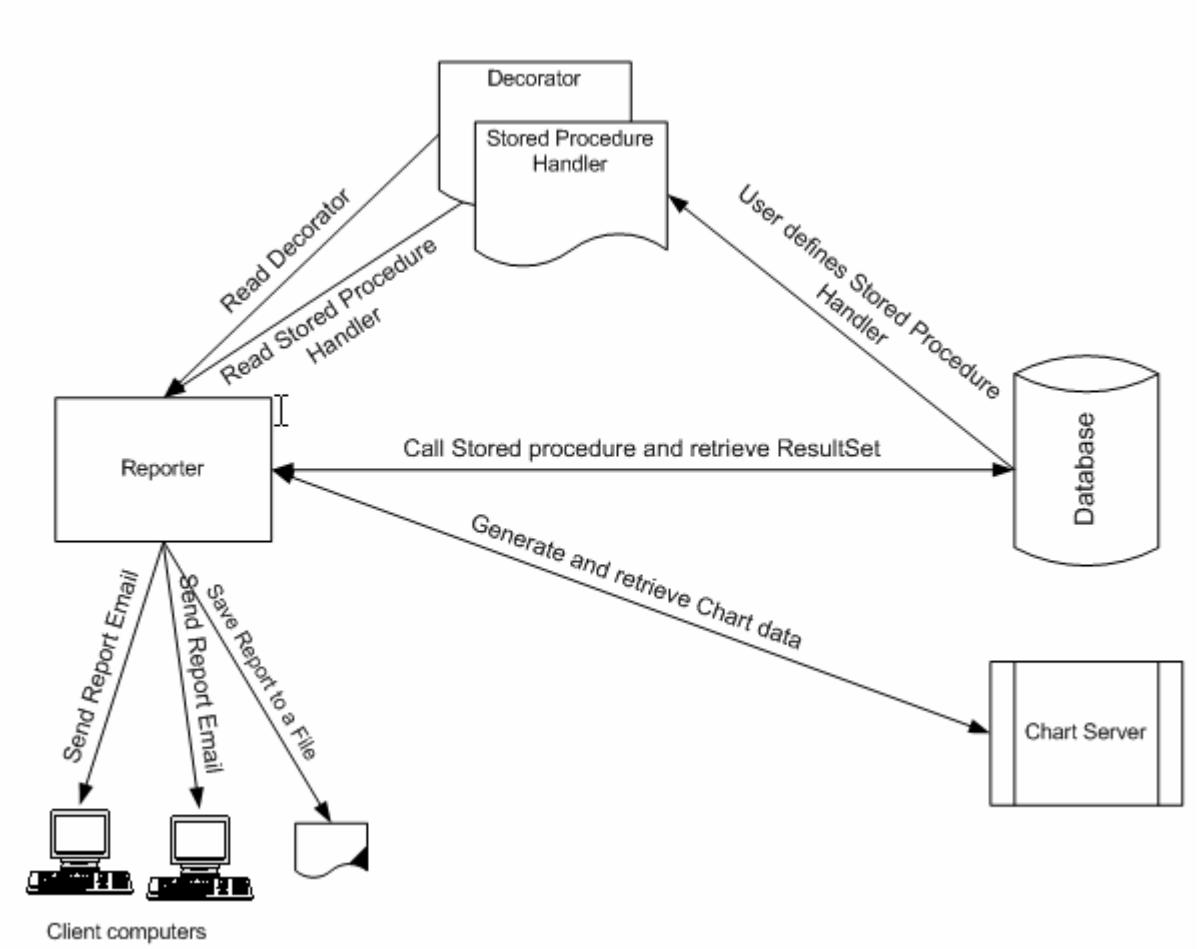
Technical Support Center Contact Information

Phone:	
Nth & Latin America	1-800-348-4357
Europe	+(44)-175-361-8000
Asia Pacific (call USA GCC)	+(1) 858-547-4765
Email:	support@hds.com

[1] Requires third party product – see Appendix C – Saving reports in PDF format

2. XML Reporting Framework Overview

2.1. Framework Architecture and Flow



The XML Reporting Framework is a standalone application that utilizes existing components from the Hitachi HiCommand® Backup Services Manager system to create custom reports. The Reporter application reads a set of configuration and descriptive files and creates reports for each of the files. Information within the files tells the Reporter where to retrieve data from the Hitachi HiCommand® Backup Services Manager data model, how to present this information and how to deliver the final report.

2.2. Decorator Files

At start up the application polls a directory containing Decorator Files. Decorator files are report descriptors that contain presentation, delivery, schedule and data retrieval information. If the file delivery schedule falls within the specified time window and the decorator is active, the reporter application starts to process the decorator. Decorator files are designed to be customized by the end-user. The decorator directory is polled on a regular basis to check for new or updated files.

2.3. Stored Procedure Handler Files

The retrieval of the data for the report is managed by a Stored Procedure Handler. The file containing the stored procedure handler is referenced in the decorator file. The stored procedure handler contains all the information required for the Reporter to programmatically call an Oracle stored procedure contained within the Hitachi HiCommand® Backup Services Manager data model. Parameters are

taken from the decorator and passed to the stored procedure, which is executed and the resulting data returned. The stored procedure handlers are supplied by Hitachi Data Systems and are not intended to be developed or customized by the end-user. See section 8.1 for a list of available Stored Procedure Handlers.

Once the data is returned the report is then formatted according to the decorator instructions. These instructions contain heading information, and the format of the report into which the data is to be placed. The decorator file can specify data to be placed into tabular, graphical, or textual formats.

Once the report is complete the decorator file specifies the delivery mechanism to be used to deliver the report, either by email or being placed into a file in a directory on the system.

2.4. Steps to Create a Report

1. The author of the custom report stored procedure (i.e. HDS) creates the stored procedure handler descriptor.
    a. This descriptor describes the IN/OUT parameters and the return cursor of the stored procedure
    b. It describes the data type of each IN/OUT parameter as well as the columns returned by the cursor
    c. It can support custom data types like NumberListType, DateType, StringType… that allow more complex data to be passed to the stored procedure.
2. The user of the XML Reporting Framework creates the report decorator file (sample decorators are contained in the samples/decorators directory and these samples can be copied and used as the basis for a new or customized decorator). The following attributes are then defined:
    a. A reference to the stored procedure handler descriptor file.
    b. Report Header
    c. Report Format
        i. HTML
        ii. PDF
        iii. Text
    d. Report type
        i. Tabular Report
        ii. Bar Chart Report
        iii. Pie Chart Report
    e. Data that needs to be displayed in the Report. The data is retrieved from references to the column parameter names as defined in the stored procedure handler descriptor.
    f. Report Totals
    g. Report Footer
    h. The schedule that the Reporter module needs to use to deliver the report
    i. The mechanism to be used to deliver the report
        i. The email distribution list
        ii. The location and file name for the saved report file
        iii. The format in which the report needs to be sent in
        iv. Report content can be chosen to attached or embedded in the email body
3. The report decorator file is then deployed to the file system (i.e. copied into the top level “decorators” directory) of the Reporter module. The report can be validated with a command line utility before being scheduled by the Reporter (see Section 4 below for details).

3. Report Components

3.1. Directories

3.1.1. Stored Procedure Handlers

`/opt/aptare/datainterface/sphandlers`

The stored procedure handlers referenced by the decorators are stored here.

3.1.2. Decorators

`/opt/aptare/datainterface/decorators`

Final Decorator files describing reports are stored here.

`/opt/aptare/datainterface/samples/decorators`

Sample decorators are stored here. These samples may be overwritten during product upgrades.

3.1.3. Templates

`/opt/aptare/datainterface/templates`

Template files referenced by the decorators are stored here.

3.1.4. Logs

`/opt/aptare/datainterface/logs`

The file reporter.log contains errors and debug information if debugging has been turned on.

3.2. The Stored Procedure Handler File

An Oracle stored procedure typically has the following structure.

A declaration of parameters being passed and returned from the procedure:

```
PROCEDURE listTapeMediaErrorSummary(
     reportingPeriod IN NUMBER := NULL,
     startTime       IN OUT NOCOPY DATE,
     finishTime      IN OUT NOCOPY DATE,
     startGroupID    IN NUMBER
     cascade         IN NUMBER
     resultCursor    OUT anyCursorType)
```

Then the body of the procedure that defines the logic and the format for the cursor being returned.

```
SELECT
   server_id, external_name, media_name, drive_name, nbr_of_mounts,
   nbr_of_restores, error_count, min_finish_time, max_finish_time,
   COUNT(DISTINCT job_id) count_events
```

The following elements of the stored procedure handler file need to be defined:

3.2.1. Stored Procedure name

Descriptive name for the file.

```
<StoredProcedure name="listTapeMediaErrorSummary">
```

3.2.2. Signature

The Oracle stored procedure name, including the Oracle package in which the stored procedure is contained. This is followed by place holders for the parameters passed in and returned.

```
<signature>custom_nbu_package.listTapeMediaErrorSummary(?,?,?,?,?,?)</signature>
```

3.2.3. Parameters

The parameters being passed and returned by the stored procedure. A parameter element contains a database parameter name, a data type and the “IN/OUT” value signifying whether the parameter is

being input to or output from the stored procedure. The values for data types and inout parameters are listed in Appendix A.

```
<Params>
    <Param dataType="2" inout="1">
        <dbParamName>reportingPeriod</dbParamName>
    </Param>
    <Param dataType="93" inout="3">
        <dbParamName>startTime</dbParamName>
    </Param>
```

3.2.4. Cursor

A special form of parameter is the cursor that is returned from a stored procedure. The cursor parameter data types and dbParamName must match the columns being returned in the cursor.

```
<Param dataType="-10" inout="2">
    <dbParamName>resultCursor</dbParamName>
    <Cursor>
        <CursorParam dataType="2">
              <dbParamName>server_id</dbParamName>
        </CursorParam>
        <CursorParam dataType="12">
              <dbParamName>external_name</dbParamName>
        </CursorParam>
```

3.3. The Data Decorator File

3.3.1. reportName

Appears in the HTML <title> tag

```
<reportName>Missing Backups Summary</reportName>
```

3.3.2. reportHeader

Appears at the top of the full report.

3.3.3. dataInterfaceActive

If this value is 0, the report decorator is ignored by the Reporter application and so the report will not execute.

0 = off, 1 = on

```
<dataInterfaceActive>0</dataInterfaceActive>
```

3.3.4. headerFile

Optional, if set to a value a file placed in the templates directory will be read and used as the initial portion of the HTML report. If not set will default to the file header.html located in the templates directiory.

```
<headerFile>custom_hdr.html</headerFile>
```

3.3.5. templateFile

If set to a value a file placed in the templates directory will be read and used to determine placement of sub reports and whether any surrounded HTML tags should be inserted into the report.

```
<templateFile/>
```

3.3.6. DataDecorator

3.3.6.1. formatType

The formatType can be one of HTML, HTMLAttachmentZipped, PDF, Delimited, Text, TextAttachment and TextAttachmentZipped. The formatType values are defined in Appendix A - Format Types.

If the report delivery mechanism is via email, then setting the formatType to AttachmentZipped will ensure the report is sent as a MIME attachment and not embedded into the email message.

A format type of Delimited will produce the data items separated by the delimiter character as defined within the file. If the delimiterType is not defined then the value defaults to "," (comma).

If this file contains DataDecorator elements such as "BarChartReport" or "PieChartReport", and the formatType is one of DELIMITED, TEXT or TextAttachment then the formatType will default to "HTML"

If the formatType is DELIMITED, TEXT or TextAttachment then there has to be at least one TableReport Decorator declared. Only TableReport Decorators are considered and processed - the other decorators are ignored.

```
<formatType>6</formatType>
```

3.3.6.2. StoredProcedureDescriptor

Reference to the stored procedure handler file to be used for this report. This file is located in the stored procedure handlers directory.

```
<StoredProcedureDescriptor>listTapeMediaErrorSummarySP.xml</StoredProcedureDes
criptor>
```

3.3.6.3. InputValues

Parameters and values to be handed to the stored procedure. The dbParamName must match the name defined in the stored procedure handler Params section. The first portion of the name must match the descriptive name in the stored procedure file.

```
<InputValues>
  <Field>
      <dbParamName>listTapeMediaErrorSummary.reportingPeriod</dbParamName>
      <value>1060</value>
  </Field>
```

3.3.6.4. TableReport

This section creates a “sub-report” within the complete report described by this decorator. The report will consist of a heading, table column headings, columns of data and optional column totals.

3.3.6.4.1. Id

Required. Needs to be unique within a decorator. Used in a template file to denote where the Report should place the report output for this table

```
<id>tableReport1</id>
```

3.3.6.4.2. StoredProcedureDescriptor

Optional. A sub-report can use a specific stored procedure handler file for its data.

3.3.6.4.3. Active

Required. A sub-report can be activated or deactivated with this flag. 1 = on, 0 = off. If deactivated, the sub-report will not execute.

```
<active>1</active>
```

3.3.6.4.4. delimiterType

Optional. If the report formatType is “Delimited”, then this value will override the default comma data delimiter.

```
<delimiterType>\t</delimiterType>
```

3.3.6.4.5. reportHeader

You can have one or more reportHeaders. An empty reportHeader represents a new line in whatever report format has been selected. You can also place input parameters in the heading by placing parameter names within braces "{".

```
<reportHeader>TAPE MEDIA ERRORS SUMMARY REPORT</reportHeader>
<reportHeader>Completed From: {listTapeMediaErrorSummary.startTime} –
{listTapeMediaErrorSummary.finishTime} </reportHeader>
```

3.3.6.4.6. dbCursorName

The stored procedure cursor that will return data for this table report.

```
<dbCursorName>listTapeMediaErrorSummary.resultCursor</dbCursorName>
```

3.3.6.4.7. tableHeaderSeparatorLine

If the report is a Text report the value of this element will be placed between the table headers and the table data.

```
<tableHeaderSeparatorLine>===============</tableHeaderSeparatorLine>
```

3.3.6.4.8. TableHeader

Specifies the parameters for a column of table data.

3.3.6.4.9. Column

A column element with the following elements is required for each piece of row data returned in the cursor.

3.3.6.4.10. dbParamName

The parameter name for values from the stored procedure cursor that will be displayed in this column.

3.3.6.4.11. displayName

Name to be used for the column heading

3.3.6.4.12. columnLength

Width of the column

3.3.6.4.13. columnJustification

Whether the column of data should be left, right or center justified. See Appendix A - Column Justification for values.

3.3.6.4.14. columnSeparator

If the report is a Text report the value of this element will be placed between the columns of data. Defaults to an empty string

3.3.6.4.15. ifNull

If the column from the database returns a NULL value the user can specify a substitute value to be displayed

3.3.6.4.16. total

If set to '1' then the Reporter will add up the column data and place the total in the last row of the table.

```
<TableHeader>
    <Column>
        <dbParamName>listTapeMediaErrorSummary.resultCursor.external_
    name</dbParamName>
        <displayName>Master Server</displayName>
        <columnLength>16</columnLength>
        <columnJustification>2</columnJustification>
    </Column>
```

3.3.6.5. BarChartReport

Optional. This section creates a "sub-report" within the complete report described by this decorator. The report will consist of a heading and an image of a bar chart report generated by the Chart Server. The bar chart consists of a number of bars and lines plotted against a X-axis and a left and right Y-axis.

3.3.6.5.1. Id

Required. Needs to be unique within a decorator. Used in a template file to denote where the Report should place the report output for this bar chart.

```
<id>barChartReport1</id>
```

3.3.6.5.2. StoredProcedureDescriptor

Optional. A sub-report can use a specific stored procedure handle file for its data.

3.3.6.5.3. Active

Required. A sub-report can be activated or deactivated with this flag. 1 = on, 0 = off.

```
<active>1</active>
```

3.3.6.5.4. chartHeight

The height in pixels of the resulting bar chart image.

```
<chartHeight>450</chartHeight>
```

3.3.6.5.5. chartWidth

The width in pixels of the resulting bar chart image.

```
<chartWidth>750</chartWidth>
```

3.3.6.5.6. leftAxisTitle

The name written against the left Y-axis.

```
<leftAxisTitle>Backup Volume</leftAxisTitle>
```

3.3.6.5.7. rightAxisTitle

The name written against the right Y-axis.

```
<rightAxisTitle># of Files Backed up</rightAxisTitle>
```

3.3.6.5.8. rightAxisIsPercentage

If set to '1' then the right axis is treated as an axis with a minimum value of 0 and a maximum of 100.

```
<rightAxisIsPercentage>1</rightAxisIsPercentage>
```

3.3.6.5.9. dbCursorName

The stored procedure cursor that will return data for this table report.

```
<dbCursorName>listJobVolumeSummary.resultCursor</dbCursorName>
```

3.3.6.5.10. xAxisDbParamName

The stored procedure parameter column name that returns X axis values. Typically dates.

```
<xAxisDbParamName>listJobVolumeSummary.resultCursor.t.the_date</xAxis
DbParamName>
```

3.3.6.5.11. xAxisDbParamGroupBy

Specifies if xAxisDbParamName is non unique. If value is "yes(1)" then, there needs to be one DataSetLabel section for each non unique value in xAxisDbParamName. This element is used to support Stacked Bar charts where a single bar contains a number of data values.

3.3.6.5.12. unitCalcDbParamName

Specifies which DbParam when encountered in the report needs to be adjusted to KB, MB, GB, and TB based on the Max value in the column.

This element allows the normalization of the value based on the Max value returned by the database. This is done especially for bar charts, when, for example, we wish to plot usage per period. If the max usage is a huge number and min usage is a small number, plotting without normalizing would make the bar for the small number invisible.

```
<unitCalcDbParamName>listJobVolumeSummary.resultCursor.n.sum_kilobyte
s</unitCalcDbParamName>
```

3.3.6.5.13. reportHeader

You can have one or more reportHeaders. An empty reportHeader represents a new line in whatever format report has been selected. You can also place input parameters in the heading by placing parameter names within braces "{".

3.3.6.5.14. DataSet

Each DataSet will be plotted as a bar on the chart,. The number of DataSets defined needs to be the same as the number of DataSet Labels

dbParamName

The parameter name for values from the stored procedure cursor that will be used to draw the bar.

```
<dbParamName>listJobVolumeSummary.resultCursor.n.sum_kilobytes</d
bParamName>
```

color

The color of the bar in HTML hexadecimal RGB codes.

```
<color>x005699</color>
```

legendStackedBar

If we have a stacked bar chart (i.e. xAxisDbParamGroupBy is yes(1)) then we can added legend values signifying the separate values in the bar.

```
<legendStackedBar>Successful</legendStackedBar>
```

Condition

A Condition Section is required if the xAxisDbParamGroupBy is yes(1). This enables “group by” for the DataSet to create the Stack in the Stack Bar.

dbParamName

The value from the stored procedure that will determine which stack this value is placed in.

value

Will currently only support a numeric data type

operator

Currently only supports the equals operator, which is the value 4. This means that if we get a value returned in the dbParamName defined under the DataSet element, then this condition is executed against another value to determine which stack this value is placed in.

```
< operator>4</operator>
```

3.3.6.5.15. DataSetLabel

The number of DataSets defined needs to be the same as the number of DataSet Labels. The label is shown as a tooltip mouseover on a bar.

label

In the case where xAxisDbParamGroupBy is "Yes(1)", then label is considered to be a string and no other "{dbParamName}" substitution string is recognized. Although the string "{columnValue}" can be specified which will be substituted with the DataSet dbParamName value. If this is the case this label is placed on the stack in a Stacked Bar chart.

```
<DataSetLabel>
<label>{listJobVolumeSummary.resultCursor.the_date}, Size:
{listJobVolumeSummary.resultCursor.sum_kilobytes},
{listJobVolumeSummary.resultCursor.sum_file_count} files backed
up.</label>
</DataSetLabel>
```

3.3.6.5.16. LineSet

The number of LineSets defined needs to be the same as the number of LineSet Labels

dbParamName

The parameter name for values from the stored procedure cursor that will be used to plot the point of the line.

```
<dbParamName>listJobVolumeSummary.resultCursor.sum_file_count</dbPara
mName>
```

If xAxisDbParamGroupBy is "Yes(1)", then the dbParam value will support a calculation of the following:

{DataSet[index].dbParamName}{operationType}{columnValue}.

Example: {DataSet0.dbParamName}{10}{columnValue}. The index for DataSet starts at 0. Only the operationType of modulus is supported. This is signified by the value 10.{columnValue} is the sum of DataSet.dbParamName grouped by xAxisDbParamName.

So if DataSet0.dbParamName = {5,10,15,20}, then the columnValue equals 5+10+15+20 or 50, then the following would be the calculated value for each LineSet.dbParamName 5%50, 10%50,15%50,20%50

color

The color of the line in HTML hexadecimal RGB codes.

3.3.6.5.17. LineSetLabel

The number of LineSets defined needs to be the same as the number of LineSet Labels

Label

This label is shown as a tool tip mouseover on a line point.

```
<label>{listJobVolumeSummary.resultCursor.t.the_date}:
{listJobVolumeSummary.resultCursor.n.sum_file_count} files backed
up.</label>
```

3.3.6.5.18. LimitLine

Draws a horizontal line across the chart

Value

No substitutors allowed here. It will ignore them if present

Label

This label is shown as a tool tip mouseover on a line point.

Color

The color of the line in HTML hexadecimal RGB codes.

3.3.6.6. PieChartReport

Optional. This section creates a “sub-report” within the complete report described by this decorator. The report will consist of a heading and an image of a pie chart report generated by the charting server.

3.3.6.6.1. Id

Required. Needs to be unique within a decorator. Used in a template file to denote where the Report should place the report output for this pie chart.

```
<id>pieChartReport1</id>
```

3.3.6.6.2. StoredProcedureDescriptor

Optional. A sub-report can use a specific stored procedure handle file for its data.

3.3.6.6.3. Active

Required. A sub-report can be activated or deactivated with this flag. 1 = on, 0 = off.

```
<active>1</active>
```

3.3.6.6.4. chartHeight

The height in pixels of the resulting pie chart image.

```
<chartHeight>450</chartHeight>
```

3.3.6.6.5. chartWidth

The width in pixels of the resulting pie chart image.

3.3.6.6.6. maxPieSlices

Max number of pie slices to display. The maxPieSlices+1 pie slice will be the aggregate total of DataSet.dbParamName for the remaining rows from the database.

```
<maxPieSlices>8</maxPieSlices>
```

3.3.6.6.7. spillOverPieSliceLabel

Label for the maxPieSlices+1 pie slice for the tooltip mouse over label

```
<spillOverPieSliceLabel>All other Servers: {spillOverData} MBytes
Used</spillOverPieSliceLabel>
```

3.3.6.6.8. dbCursorName

The stored procedure cursor that will return data for this pie chart report.

3.3.6.6.9. legendDbParamName

A pie chart has a legend drawn beneath it with a colored square and a label relating to the pie slices. The value from the cursor will be used as the legend.

```
<legendDbParamName>listClientUsageSummary.resultCursor.external_name<
/legendDbParamName>
```

3.3.6.6.10. spillOverLegendName

```
<spillOverLegendName>All Other Servers</ spillOverLegendName>
```

3.3.6.6.11. unitCalcDbParamName

Specifies which DbParam when encountered in the report needs to be adjusted to KB, MB, GB, and TB based on the Max value in the column.

See section 3.3.6.5.12 above

3.3.6.6.12. reportHeader

You can have one or more reportHeaders. An empty reportHeader represents a new line in whatever format report has been selected. You can also place input parameters in the heading by placing parameter names within braces "{".

3.3.6.6.13. DataSet

dbParamName

The parameter name for values from the stored procedure cursor that will be used to draw the pie.

```
<dbParamName>listClientUsageSummary.resultCursor.sum_tape_mbytes</dbP
aramName>
```

label

This label is shown as a tool tip mouseover on a pie slice.

```
<label>{listClientUsageSummary.resultCursor.external_name}:
{listClientUsageSummary.resultCursor.sum_tape_mbytes} MBytes
Used</label>
```

color

This label color of the pie slice. The number of colors must match the value of maxPieSlices

```
<color>#a0bdc4,#acc86c,#aa0036,#ffc800,#005699,#5B7C0F,#f3e68c,#F5990
9</color>
```

OperatorDbParamName

Performs a mathematical operation on {DataSet.dbParamName} and {OperatorDbParamName.dbParamName} producing the value for {DataSet.dbParamName}. {DataSet.dbParamName} = {DataSet.dbParamName} OPERATOR {OperatorDbParamName.dbParamName}

Currently we support operator "3" which is "/" the "divide" operator. The tag {total} represents SUM(DataSet.dbParamName)

```
<OperatorDbParamName operator="3">
    <dbParamName>{total}</dbParamName>
</OperatorDbParamName>
```

3.3.7. Delivery

Note that both File and Mail sections can be specified to send an email and create a file.

3.3.7.1. Schedule

Schedule follows a Unix "cron" type scheduling convention. There are five categories that represent a schedule. The categories are minute, hour, day of month, month, day of week. The categories are separated by spaces. An asterisk ('*') signifies that this category represents all available values. You can specify specific values by separating values by commas. And you can specify that a category represents a repeating range with a forward slash.

All the indices start at 0, except day of month and month that begin with 1. Minute has a range of 0-59, hour a range of 0-23, day of month a range of 1-31, month a range of 1-12 and day of week a range of 0-6 (0 is Sunday).

Examples:

```
<schedule>0 1 * * *</schedule> <!-- 1am every day -->
```

Each minute

* * * * *

Each 5 minutes

*/5 * * * *

Every 10 minutes every two hours

*/10 */2 * * *

At half of odd hours

30 1,3,5,7,9,11,13,15,17,19,21,23 * * *

Every second hour of every second month

0 */2 * */2 *

Mondays at 20:00h

0 20 * * 1

Sundays at 20:00h

0 20 * * 0

5th day of each month at 00:00h

0 0 5 * *

3.3.7.2. File

3.3.7.2.1. Location

The location on the server where the reporter application is running where files will be written to. It is the responsibility of the administrator of the system to ensure files are removed after they have been viewed.

```
<location>/opt/aptare/datainterface/reports/</location>
```

3.3.7.2.2. Name

The filename of the report being placed in the location directory.

```
<name>ClientUsageSummary - {currentDate}</name>
```

3.3.7.3. Mail

The configuration file for mail server settings is found at the default location: /opt/aptare/datainterface/conf/datainterfaceconfig.xml

If Format Type is HTML, then the report will be sent in the body of the mail.

If Format Type is PDF, then the report will be sent as an attachment

If Format Type is DELIMITED, then the report will be sent as an attachment

If Format Type is TEXT, then the report will sent as the body of the mail

If Format Type is TEXTAttachment, then the report will be sent as an attachment

3.3.7.3.1. toMailIds

The report will be mailed to the following mail addresses. Multiple addresses are separated by commas “,”.

```
<toMailIds>user@yourdomain.com</toMailIds>
```

3.3.7.3.2. fromMailId

Most mail servers require a valid mail address to be used when mail is being sent.

```
<fromMailId>user@yourdomain.com</fromMailId>
```

3.4. The template files

3.4.1. Header file

The file (header.html) in the templates directory is pre-pended to all HTML reports. You can override this with the use of the headerFile tag. This file contains the initial portion of the HTML page and includes the Cascading Style Sheet used by all the reports and JavaScript that enables dynamic table column sorting.

3.4.2. Template file

If a decorator file references a template file, this file should be placed in the templates directory. Sub-report Id values are placed in this file and can be surrounded by HTML tags. Each report is contained with a HTML table section.

```
tableReport1
<p/>
barChartReport1
```

3.4.3. Footer file

The file (footer.html) in the templates directory is appended to all HTML reports

4. Testing the Report

You are now ready to run a simple test to verify that the reports run and are emailed successfully to the list of recipients that you configured. The following command line programs can be found in /opt/aptare/datainterface/bin

For UNIX platforms:

```
./validateDecorator.sh listTapeMediaErrorSummary.xml
```

For Windows platforms, open a command window and type

```
validateDecorator.bat listTapeMediaErrorSummary.xml
```

5. Modifying the Look And Feel of HTML reports

5.1. Sections of a Report

By default the report that is produced contains five sections. The first, produced by the header file, contains all the required HTML, including the CSS and the JavaScript, for the report up to the <BODY> tag. If you wish to modify META tags, add your own CSS classes or modify existing classes, you should make a copy of the existing header, modify this copy then override the default file with the use of a headerFile tag in the decorator.

The second section is the overall report heading. This is a self contained HTML table generated from information within the decorator.

Each sub-report referenced in the decorator contains two sections. The first is a self contained HTML table containing the sub report heading generated from information resident in the decorator. The second section contains the report itself. A HTML table in the event of a tableReport, an image and image map in the event of a Bar Chart, and for a Pie Chart, an image, imagemap and HTML table representing the pie slice legend.

The last section is the footer information containing copyright information and the closing BODY and HTML tags.

5.2. Modifying the Styles

Every HTML tag generated by the Reporter comes with a CSS class that can be modified with a custom header file. For instance, by default every sub report header has the following CSS style.

```
table.tableReportHeader, table.barReportHeader, table.pieReportHeader {
    width : 100%;
    border-bottom : #020203 1px solid;
    border-left : #020203 1px solid;
    border-right : #020203 1px solid;
    border-top : #020203 1px solid;
```

with the following HTML generated

```
<!-- sub-Header Section -->
<table class="barReportHeader">
<tr>
   <td class="header">
      <nobr><h2 class="reportHeader">Bar Chart Showing XYZ</h2>
      </nobr>
   </td>
</tr>
</table>
<!-- END sub-Header Section -->
```

which draws a box around the heading information. You can modify the CSS so that the barReportHeader class does not draw a line around the heading with the following code:

```
table.barReportHeader {
    width : 100%;
    border-bottom : medium none;
    border-left : medium none;
    border-right : medium none;
    border-top : medium none;
}
```

6. Example

6.1. Decorator Volume and Errors File

```
<?xml version="1.0" encoding="UTF-8"?>
<APTAREDataInterface xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:noNamespaceSchemaLocation="C:\Program Files\Apache Group\Tomcat
4.1\webapps\portal\conf\APTAREDataInterface.xsd">
    <reportName>Consecutive Errors by Client and Backup Volume Summary</reportName>
    <reportHeader>Consecutive Errors by Client and Backup Volume Summary</reportHeader>
    <dataInterfaceActive>0</dataInterfaceActive>
    <templateFile>volumeanderrors.html</templateFile>
    <DataDecorator>
        <formatType>1</formatType>
        <InputValues>
            <Field>
                <dbParamName>listJobVolumeSummary.dateRange</dbParamName>
                <!--daterangetype -
                timePeriod,startTime,finishTime,startDate,finishDate,timezoneName,useFinishTim
                e -->
                <value>1021,null,null,null,null,null,1</value>
            </Field>
            <Field>
                <dbParamName>listJobVolumeSummary.startGroupID</dbParamName>
                <value>100000</value>
            </Field>
            <Field>
                <dbParamName>listJobVolumeSummary.clientIDList</dbParamName>
                <value>null</value>
            </Field>
            <Field>
                <dbParamName>listJobVolumeSummary.cascade</dbParamName>
```

```
                    <value>1</value>
                </Field>
                <Field>
                    <dbParamName>listJobVolumeSummary.ignoreRetries</dbParamName>
                    <value>1</value>
                </Field>
                <Field>
                    <dbParamName>listJobVolumeSummary.jobTypeList</dbParamName>
                    <value>1,2,3,4,5,6</value>
                </Field>
                <Field>
                    <dbParamName>listJobVolumeSummary.jobStatusList</dbParamName>
                    <value>0,1,2,3,4,5,6,7,8,9</value>
                </Field>
                <Field>
                    <dbParamName>listJobVolumeSummary.groupBy</dbParamName>
                    <value>11</value>
                </Field>
                <Field>
                    <dbParamName>listConsecutiveBackupErrors.dateRange</dbParamName>
                    <!--daterangetype -
                    timePeriod,startTime,finishTime,startDate,finishDate,timezoneName,useFinishTim
                    e -->
                    <value>1021,null,null,null,null,null,1</value>
                </Field>
                <Field>
                    <dbParamName>listConsecutiveBackupErrors.groupID</dbParamName>
                    <value>0</value>
                </Field>
                <Field>
                    <dbParamName>listConsecutiveBackupErrors.cascade</dbParamName>
                    <value>1</value>
                </Field>
                <Field>
                    <dbParamName>listConsecutiveBackupErrors.clientList</dbParamName>
                    <value>null</value>
                </Field>
                <Field>
                    <dbParamName>listConsecutiveBackupErrors.jobTypeList</dbParamName>
                    <value>1,2,3,4,5,6</value>
                </Field>
                <Field>
                    <dbParamName>listConsecutiveBackupErrors.excludeStatusList</dbParamName>
                    <value>null</value>
                </Field>
                <Field>
                    <dbParamName>listConsecutiveBackupErrors.nbrOfConsecErrors</dbParamName>
                    <value>2</value>
                </Field>
            </InputValues>
            <TableReport>
                <id>tableReport1</id>

        <StoredProcedureDescriptor>listConsecutiveBackupErrorsSP.xml</StoredProcedureDescriptor>
                <active>1</active>
                <delimiterType>\t</delimiterType>
                <reportHeader> </reportHeader>
                <reportHeader> </reportHeader>
                <reportHeader> </reportHeader>
                <reportHeader>       C O N S E C U T I V E   E R R O R S   B Y   C L I E N T
                R E P O R T </reportHeader>
                <reportHeader>             Completed From
                {listJobVolumeSummary.dateRange.startDate} -
                {listJobVolumeSummary.dateRange.finishDate} </reportHeader>
                <reportHeader>                              Sorted By Error Count </reportHeader>
                <reportHeader> </reportHeader>
                <reportHeader> </reportHeader>
                <dbCursorName>listConsecutiveBackupErrors.resultCursor</dbCursorName>

            <tableHeaderSeparatorLine>================================================================
        </tableHeaderSeparatorLine>
                <TableHeader>
```

```
                    <Column>
                          <dbParamName>listMissingBackupsByClient.resultCursor.s.external_name</dbP
                          aramName>
                            <displayName>Client Hostname</displayName>
                            <columnLength>30</columnLength>
                            <columnJustification>2</columnJustification>
                    </Column>

                    <Column>
                          <dbParamName>listMissingBackupsByClient.resultCursor.ms.external_name</db
                          ParamName>
                            <displayName>Master Hostname</displayName>
                            <columnLength>20</columnLength>
                            <columnJustification>3</columnJustification>
                    </Column>
                    <Column>
                          <dbParamName>listMissingBackupsByClient.resultCursor.consecutive_errors</
                          dbParamName>
                            <displayName>Errors</displayName>
                            <columnLength>20</columnLength>
                            <columnJustification>3</columnJustification>
                    </Column>
                    <Column>
                          <dbParamName>listMissingBackupsByClient.resultCursor.consecutive_days</db
                          ParamName>
                            <displayName>Days</displayName>
                            <columnLength>20</columnLength>
                            <columnJustification>3</columnJustification>
                    </Column>
                    <Column>
                          <dbParamName>listMissingBackupsByClient.resultCursor.first_error_date</db
                          ParamName>
                            <displayName>First Error Date</displayName>
                            <columnLength>20</columnLength>
                            <columnJustification>3</columnJustification>
                    </Column>
                    <Column>
                          <dbParamName>listMissingBackupsByClient.resultCursor.last_error_date</dbP
                          aramName>
                            <displayName>Last Error Date</displayName>
                            <columnLength>20</columnLength>
                            <columnJustification>3</columnJustification>
                    </Column>
                </TableHeader>
                <reportFooter/>
            </TableReport>
            <BarChartReport>
                <id>barChartReport1</id>
                <StoredProcedureDescriptor>listJobVolumeSummarySP.xml</StoredProcedureDescriptor>
                <active>1</active>
                <chartHeight>450</chartHeight>
                <chartWidth>750</chartWidth>
                <leftAxisTitle>Backup Volume</leftAxisTitle>
                <rightAxisTitle># of Files Backed up</rightAxisTitle>
                <dbCursorName>listJobVolumeSummary.resultCursor</dbCursorName>
                <xAxisDbParamName>listJobVolumeSummary.resultCursor.t.the_date</xAxisDbParamName>

        <unitCalcDbParamName>listJobVolumeSummary.resultCursor.n.sum_kilobytes</unitCalcDbParamNa
    me>
                <reportHeader> </reportHeader>
                <reportHeader> </reportHeader>
                <reportHeader> </reportHeader>
                <reportHeader>        B A C K U P   V O L U M E   S U M M A R Y   R E P O R T
    </reportHeader>
                <reportHeader>              Completed From
    {listJobVolumeSummary.dateRange.startDate} - {listJobVolumeSummary.dateRange.finishDate}
    </reportHeader>
                <reportHeader> </reportHeader>
                <reportHeader> </reportHeader>
                <DataSet>
                    <dbParamName>listJobVolumeSummary.resultCursor.n.sum_kilobytes</dbParamName>
                    <color>x005699</color>
```

```
            </DataSet>
            <DataSetLabel>
                <label>{listJobVolumeSummary.resultCursor.t.the_date}, Size:
{listJobVolumeSummary.resultCursor.n.sum_kilobytes},
{listJobVolumeSummary.resultCursor.n.sum_file_count} files backed up.</label>
            </DataSetLabel>
            <LineSet>
                <dbParamName>listJobVolumeSummary.resultCursor.n.sum_file_count</dbParamName>
                <color>x666666</color>
            </LineSet>
            <LineSetLabel>
                <label>{listJobVolumeSummary.resultCursor.t.the_date}:
{listJobVolumeSummary.resultCursor.n.sum_file_count} files backed up.</label>
            </LineSetLabel>
            <reportFooter/>
        </BarChartReport>
    </DataDecorator>
    <Delivery>
        <schedule>*/5 * * * *</schedule>
        <!--Comment out file or mail section to deliver via mail or create file repectively-->
        <!--<file>
            <location>/opt/aptare/datainterface/reports/</location>
            <name>BackupVolumeAndConsecutiveErrors - {currentDate}</name>
        </file>-->
        <mail>
            <toMailIds>user@yourdomain.com</toMailIds>
            <fromMailId>user@yourdomain.com</fromMailId>
        </mail>
    </Delivery>
</APTAREDataInterface>
```

6.2. Output

Consecutive Errors by Client and Backup Volume Summary

CONSECUTIVE ERRORS BY CLIENT REPORT

Completed From Jan 13, 2006 12:00:00AM - Feb 02, 2006 04:51:27PM

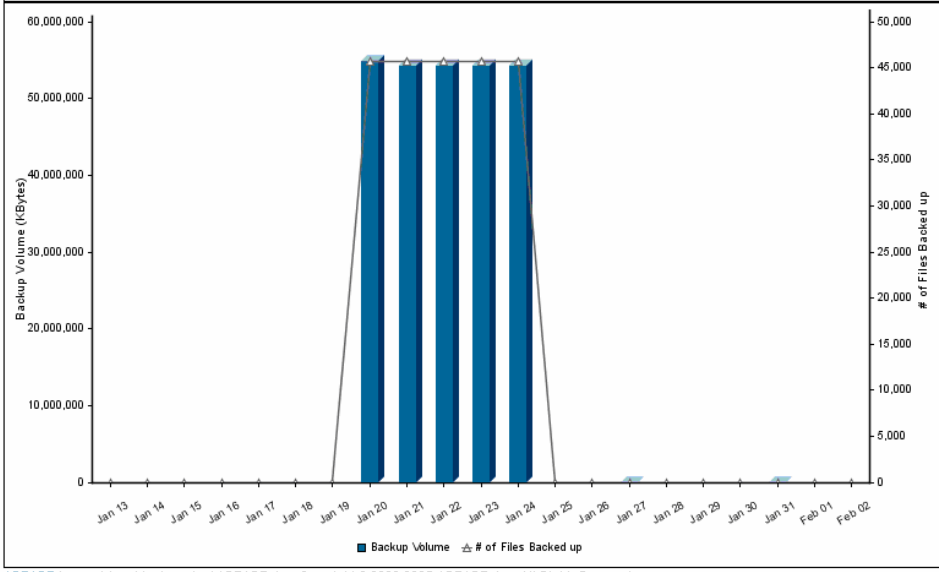
Sorted By Error Count

Client Hostname	Master Hostname	Errors	Days	First Error Date	Last Error Date
aptareprod1	aptareprod1	4	2	Feb 01, 2006 01:04:22am	Feb 02, 2006 01:14:22am
aptaresqa	aptareprod1	3	1	Feb 01, 2006 12:04:28am	Feb 02, 2006 12:04:27am
aptareweb1	aptareprod1	2	1	Feb 02, 2006 01:04:21am	Feb 02, 2006 01:14:22am
aptaredev1	aptareprod1	2	1	Feb 01, 2006 12:04:28am	Feb 02, 2006 12:04:27am
aptaredev3	aptareprod1	2	1	Feb 01, 2006 12:34:28am	Feb 02, 2006 12:34:27am

Total Rows: 5

BACKUP VOLUME SUMMARY REPORT

Completed From Jan 13, 2006 12:00:00AM - Feb 02, 2006 04:51:27PM



7. Appendix A – Parameter types and values

7.1. Parameter Data Type Values

Data Type	Value
LONGVARCHAR	-1
BINARY	-2
VARBINARY	-3
LONGVARBINARY	-4
BIGINT	-5
TINYINT	-6
BIT	-7
CURSOR	-10
NULL	0
CHAR	1
NUMERIC	2
DECIMAL	3
INTEGER	4
SMALLINT	5
FLOAT	6
REAL	7
DOUBLE	8
VARCHAR	12
DATE	91
TIME	92
TIMESTAMP	93
dateRangeType	200
numberListType	201
APTSTATUSCODELIST	202
backupWindowListType	203

Where

dateRangeType=timePeriod,startTime,finishTime,startDate,finishDate,timezoneName,useFinishTime

```
<value>1021,null,null,null,null,null,1</value>
```

See Appendix A - Time Period and Reporting Period Constants for valid timePeriod values.

numberListType = comma separated list of numbers

```
<value>1,2,3,4,5,6</value>
```

backupWindowListType = set of backup window values, with Window name, the window start in hours since Sunday and window end in hours since Sunday.

```
<value>(Monday, 40, 64) (Tuesday, 64, 88) (Wednesday, 88, 112) (Thursday, 112, 136)
(Friday, 136, 160) (Saturday, 160, 184) (Sunday, 16, 40)</value>

(Monday, 40, 64)     -- Mon 16:00 -> Tue 16:00
(Tuesday, 64, 88)    -- Tue 16:00 -> Wed 16:00
(Wednesday, 88, 112) -- Wed 16:00 -> Thu 16:00
(Thursday, 112, 136) -- Thu 16:00 -> Fri 16:00
(Friday, 136, 160)   -- Fri 16:00 -> Sat 16:00
(Saturday, 160, 184) -- Sat 16:00 -> Sun 16:00
```

Timestamp = yyyy-mm-dd hh:mm:ss.fffffffff

```
<value>2005-10-1 10:00:00</value>
```

7.2. Format Types

Format Type	Value
HTML	1
HTMLAttachmentZipped	3
PDF	4
Delimited	5
Text	6
TextAttachment	7
TextAttachmentZipped	8

7.3. Time Period and Reporting Period Constants

Period	Value
LAST_12_HOURS	1
LAST_24_HOURS	2
LAST_48_HOURS	3
LAST_72_HOURS	4
WEEK_TO_DATE	5
MTH_TO_DATE	6
QTR_TO_DATE	7
YEAR_TO_DATE	8
PREV_WEEK	9
PREV_MTH	10
PREV_QTR	11
PREV_YEAR	12

LAST_7_DAYS	1007
LAST_10_DAYS	1010
LAST_14_DAYS	1014
LAST_21_DAYS	1021
LAST_30_DAYS	1030
LAST_60_DAYS	1060
LAST_120_DAYS	1120
LAST_180_DAYS	1180
LAST_365_DAYS	1365
LAST_YEAR	2001
LAST_3_YEARS	2003
LAST_5_YEARS	2005
LAST_7_YEARS	2007

7.4. Column Justification

Justification	Value
Right Justified	1
Left Justified	2
Center Justified	3

8. Appendix B – Stored Procedure Handlers and Decorators

8.1. Available Stored Procedure Handlers

Stored Procedure Handler File	Stored Procedure Signature
listClientUsageSummarySP.xml	**Stored Procedure Name:** all_usage_pkg.listClientUsageSummary **Input Parameters:** dateRange startGroupID - ID of the group to start cascading cascade - Cascade the server groups if set to true (i.e. 1) clientList - List of client/server IDs jobTypeList- List of job types **Output Parameters:** resultCursor(client_id, internal_name, external_name, ip_address, server_id, server_internal_name, server_external_name, server_ip_address, product_type_name, sum_tape_mbytes, sum_disk_mbytes, total_tape_files, total_disk_files, total_tapes, job_id, max_finish_date)
listConsecutiveBackupErrorsSP.xml	**Stored Procedure Name:** all_error_pkg.listConsecutiveBackupErrors **Input Parameters:** dateRange groupID - ID of the group to start cascading cascade - Cascade the server groups if set to true (i.e. 1) clientList - List of client/server IDs jobTypeList - NULL indicates all job types excludeStatusList nbrOfConsecErrors **Output Parameters:** resultCursor (t.client_id, s.internal_name, s.external_name, s.ip_address,t.server_id, ms.internal_name, ms.external_name, first_error_id, first_error_date, last_error_id,last_error_date,consecutive_days, consecutive_errors)
listJobVolumeSummarySP.xml	**Stored Procedure Name:** all_report_pkg.listJobVolumeSummary **Input Parameters:** dateRange, startGroupID, clientIDList, cascade, ignoreRetries, jobTypeList, jobStatusList, groupBy **Output Parameters:** resultCursor(t.the_date, n.sum_jobs,

	n.sum_kilobytes, n.sum_file_count)
listMissingBackupsByClient.xml	**Stored Procedure Name:** custom_nbu_package.listMissingBackupsByClient **Input Parameters:** reportingPeriod, startDate, finishDate, startGroupID, cascade **Output Parameters:** resultCursor(serverId, internalName, finishDate)
listTapeMediaUsageSP.xml	**Stored Procedure Name:** all_tape_pkg.listTapeMediaUsage **Input Parameters:** dateRange, startGroupID, cascade, clientIDList, vendorMediaType, groupBy **Output Parameters:** resultCursor(t.the_date, t1.tape_count)
listTapeMediaErrorSummarySP.xml	**Stored Procedure Name:** custom_nbu_package.listTapeMediaErrorSummary **Input Parameters:** reportingPeriod, startDate, finishDate, startGroupID, cascade **Output Parameters:** resultCursor(t1.server_id, t1.external_name, t1.media_name, d.drive_name, t1.nbr_of_mounts, t1.nbr_of_restores, t1.error_count, t1.min_finish_time, t1.max_finish_time, count_events)
listSLASummarySP.xml	**Stored Procedure Name:** xml_report_pkg.listSLASummary **Input Parameters:** dateRange, startGroupID, cascade, clientIDList, ignoreRetries, jobTypeList, jobStatusList, groupBy, backupObjective **Output Parameters:** resultCursor(t.the_date, the_date, the_status, the_value, success_pct, failure_pct)
listSummaryTotalsByClient.xml	**Stored Procedure Name:** xml_nbu_report_pkg.listSummaryTotalsByClient **Input Parameters:** dateRange, backupWindowList, groupID,cascade, clientList, serverList, excludeStatusList, byManagementServer **Output Parameters:** external_name, start_date, finish_date, total_clients, nbr_successful, count_error, count_running, count_queued, pct_success
listSuspectClientBackups.xml	**Stored Procedure Name:** xml_nbu_report_pkg.listSuspectClientBackups

	Input Parameters: reportingPeriod, startDate, finishDate, startGroupID, variancePct **Output Parameters:** client_hostname, pathname, policy_schedule_name, schedule_type_name, start_date, finish_date, megabytes, avg_megabytes
listTapeStatusSummarySP.xml	**Stored Procedure Name:** all_tape_pkg.listTapeStatusSummary **Input Parameters:** userHomeGroupID, productType, startGroupID, cascade, serverIDList, clientIDList, libraryList **Output Parameters:** resultCursor(product_type, vendor_media_status, media_status_name, tape_count)
listClientUsageSummaryDiskTapeSP.xml	**Stored Procedure Name:** xml_report_pkg.listClientUsageSummary **Input Parameters:** dateRange, startGroupID, cascade, clientIDList, jobTypeList **Output Parameters:** resultCursor(the_string, the_number)

8.2. Available Report Decorators

Decorator File	Report Title	Stored Procedure Handler File	Template File
listClientUsageSummary.xml	Client Tape Usage Report	listClientUsageSummarySP.xml	
listConsecutiveBackupErrors.xml	Consecutive Errors By Client	listConsecutiveBackupErrorsSP.xml	
listJobVolumeSummary.xml	Backup Volume Summary Report	listJobVolumeSummarySP.xml	
listMissingBackupsByClient.xml	Missing Backups Summary	listMissingBackupsByClient.xml	
listTapeMediaUsage.xml	Tape Media Usage Report	listTapeMediaUsageSP.xml	
listTapeMediaErrorSummary.xml	Tape Media Errors Summary	listTapeMediaErrorSummarySP.xml	
listSuspectClientBackups.xml	List Suspect Client Backups	listSuspectClientBackups SP.xml	
listSummaryTotalsByClient.xml	Client Backup SLA Summary Report	listSummaryTotalsByClient SP.xml	
volumeanderrors.xml	Consecutive Errors by Client and Backup Volume Summary	listConsecutiveBackupErrorsSP.xml listJobVolumeSummarySP.xml	volumeanderrors.html
op_dashboard.xml	Operations Dashboard	istSLASummarySP.xml listTapeMediaUsageSP.xml listTapeStatusSummarySP.xml listClientUsageSummaryDiskTapeSP.xml	op_dashboard.html

9. Appendix C – Saving reports in PDF format

To output reports to a PDF format, a third party library is required. The instructions on how to obtain and install this library are detailed below.

1. Purchase and download "PD4ML Java Library Standard" from http://pd4ml.com
2. Extract the file "pd4ml.jar" from the downloaded PD4ML Java Library from step 1.
3. Copy the file "pd4ml.jar" into $APTARE_HOME/lib on the portal server (e.g. /opt/aptare/lib)
4. For a Windows Portal Server:
    1. Edit $APTARE_HOME/datainterface/conf/reporterwrapper.conf and insert the line "wrapper.java.classpath.26=../../lib/pd4ml.jar" right after the line "wrapper.java.classpath.25=../../lib/jcrontab.jar"
    2. Edit the file $APTARE_HOME/datainterface/bin/validateDecorator.bat, and insert ";c:\opt\aptare\lib\pd4ml.jar" right after "c:\opt\aptare\lib\jargs.jar"
5. For a UNIX Portal Server:
    1. Edit $APTARE_HOME/datainterface/bin/startReporterEngine, and insert ":$APTARE_HOME/lib/pd4ml.jar" right after "$APTARE_HOME/lib/jargs.jar"
    2. Edit $APTARE_HOME/datainterface/bin/validateDecorator.sh, and insert ":$APTARE_HOME/lib/pd4ml.jar" right after "$APTARE_HOME/lib/jargs.jar"
6. Restart the APTARE XML Reporting Service via the Windows Services control panel or via "/opt/aptare/bin/aptare-reporter restart" on UNIX.