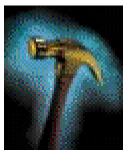


Deploying Interactive Charts on the Web: Migrating from the Graphics Cartridge

An Oracle Technical White Paper
October 2000

INTRODUCTION

This paper is intended for developers who are looking for an alternative way of deploying Oracle Graphics displays over the Web now that the Oracle Graphics Cartridge is being desupported. The alternative is to embed the display within an Oracle Report deployed over the web via the Oracle Reports Server.

For non-interactive Oracle Graphics redeployment is simply a matter of creating a chart item in a report displayed over the Web. For interactive Oracle Graphics the situation is more complex: displays which currently use button procedures or have drill-down properties set on chart items will require modifications if they are to achieve this interactive behavior through the Reports Server.

This paper explains how to migrate interactive graphics displays from Graphics Cartridge to Reports Server deployment and the options available in doing so.

INTERACTIVE CHARTS ON THE WEB

Currently, there are four ways to deploy Oracle Graphics on the Web:

- Embed them in Oracle Forms.
- Embed them in Oracle Reports.
- Use the Graphics Cartridge.
- Export them as image files embedded in a Web page.

Up until Developer 6.0, interactive Oracle Graphics could be deployed using either the OG.PLL package within Oracle Forms or the Graphics Cartridge. With Developer 6*i* Release 2, the Graphics Cartridge has been desupported. However, with Developer 6.0 new functionality was added to the Reports Server to allow embedded charts to be interactive. Although neither Oracle Graphics button procedures nor drill-down properties set on chart objects will work on charts deployed through the Reports Server, two methods of enabling interactive charting are available:

- Chart Hyperlinks
- The OG_SET_URL built-in

Chart Hyperlinks allow the specification of an individual URL for any chart object which represents a data value (a bar, a pie slice, but not an axis or grid), provided the chart query data is passed to Oracle

Graphics from a report query. The new OG_SET_URL built-in allows a dynamic URL to be set on any object in an Oracle Graphics display.

The Reports Server is the recommended option for anyone migrating static or interactive charts from the Graphics Cartridge. It will give the best results and the best performance—Reports Server deployment has been found to demand considerably less memory and CPU resources than the Graphics Cartridge. All the drill-down behavior which can be achieved with Graphics displays deployed using the Graphics Cartridge can now be achieved over the Web from within Oracle Reports, though in a different way.

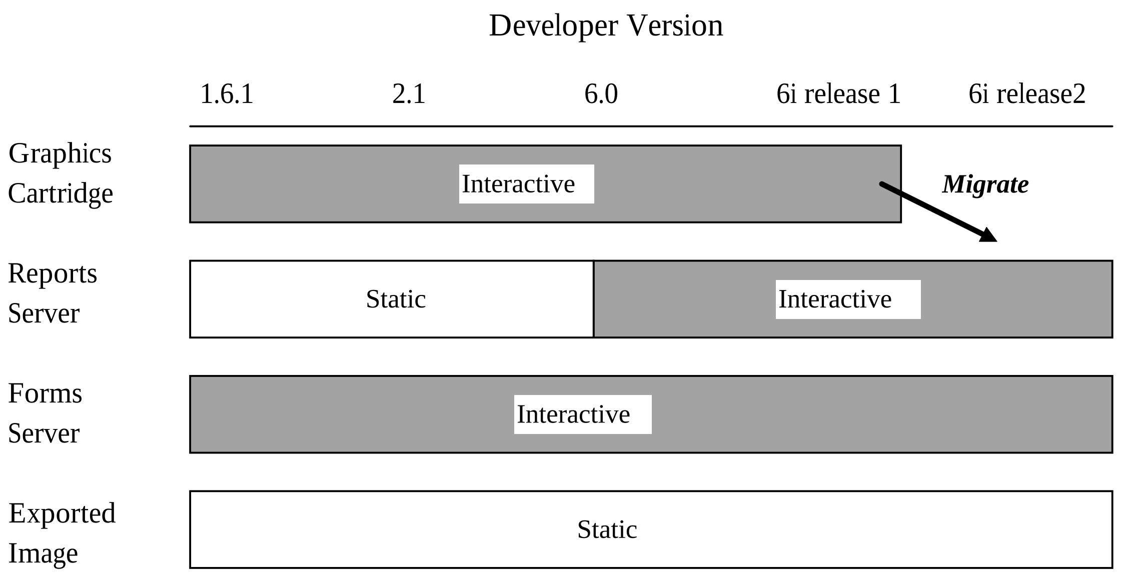


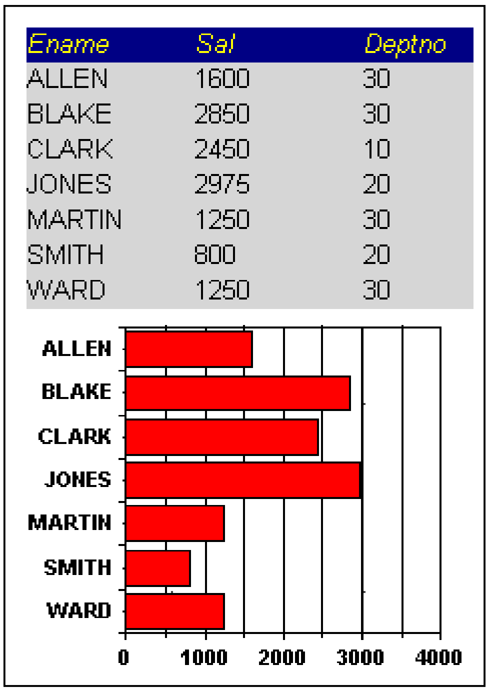
Figure 1. Static and interactive charting on the Web in different versions of Oracle Developer

CHART HYPERLINKS

If Reports passes a query to an embedded chart, it is able to map each chart object representing the data back to its associated row in the report query. Click on the chart object, and Reports can reference any of the data items on that row in the query. For example, Figure 2 illustrates a report based on the query

```
SELECT deptno, ename, sal FROM emp
```

with an embedded chart which represents ename and sal.



Ename	Sal	Deptno
ALLEN	1600	30
BLAKE	2850	30
CLARK	2450	10
JONES	2975	20
MARTIN	1250	30
SMITH	800	20
WARD	1250	30

Figure 2. A report query also displayed in a chart

Click on the bar representing Smith's salary, and Reports can access his deptno, ename, and sal. This is possible because the chart item on the report has a property *Chart Hyperlink*. Here the values of these columns can be referenced as &deptno, &ename, and &sal, and concatenated into a hyperlink which is called when the bar is selected. To go from this bar to another Web page containing details on Smith's department, taking his deptno as a parameter, the chart hyperlink might look something like this:

```
http://myserv.mycomp.com/deptarmentdetails.html+dept=&deptno
```

USING CHART HYPERLINKS TO MIGRATE FROM THE GRAPHICS CARTRIDGE

For many interactive charts, chart hyperlinks are the simplest option for migrating from the Graphics Cartridge to the Reports Server. Consider a typical example: Figure 3 illustrates a master-detail pair of charts that uses a pie chart to represent total departmental salaries and a column chart to represent individual salaries for a particular department. Clicking on a particular pie slice, for example *Accounting*, brings up the bar chart for that department.

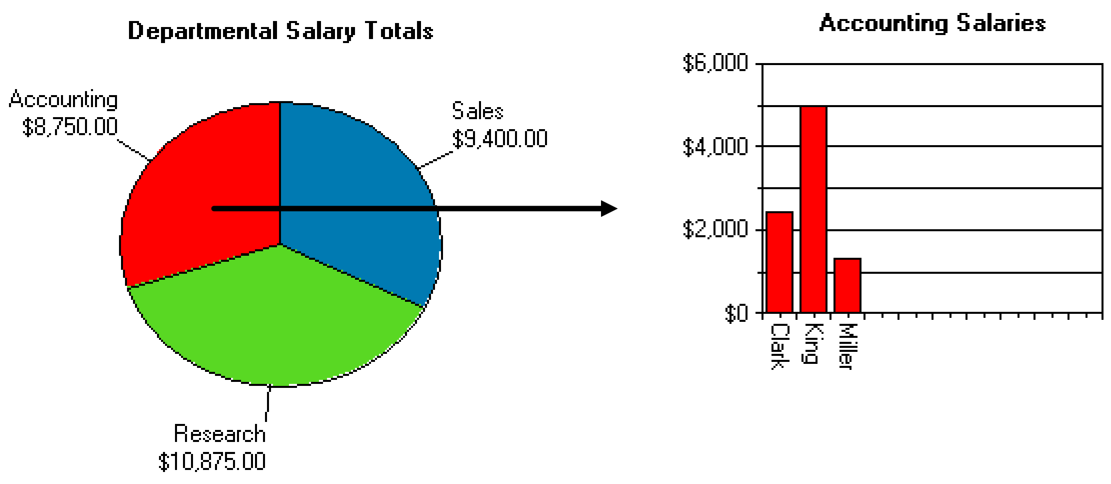


Figure 3. Master and Detail Charts

If the two charts were contained in different displays, the normal method in Graphics would be to use a button procedure on the pie slice to call the other chart and pass the appropriate key value. Using this example, migrating to the Reports server using Chart Hyperlinks would involve four stages:

1. Embed the charts in Reports.
2. Set up Reports parameters corresponding to the Oracle Graphics parameters.
3. Move the query of the interactive chart into Reports
4. Set the Chart Hyperlink property.

The following sections describe these stages in detail.

1. Embed the charts in Reports

Embed each chart in a separate report. Each report requires nothing other than the chart object on its layout.

Embedding is straightforward:

1. Open the Reports Builder, and select *Build a new report manually.*
2. Bring up the Layout Editor, and create a chart item.
3. Open the Property Palette of the chart item, and enter the chart filename.

2. Set up Reports parameters corresponding to Oracle Graphics parameters

The detail chart, and possibly also the master chart, will require parameters to be passed to them.

1. Under the *Data Model* node in the Reports Object Navigator, create user parameters corresponding to each of the parameters in the embedded OGD.
2. Open the Property Palette of the chart item, and select *Parameters and Columns.*
3. Highlight each parameter you created in Reports, and enter the corresponding chart parameter alongside it.

For example, in Figure 4 the report parameter deptno is passed to an OGD parameter of the same name:

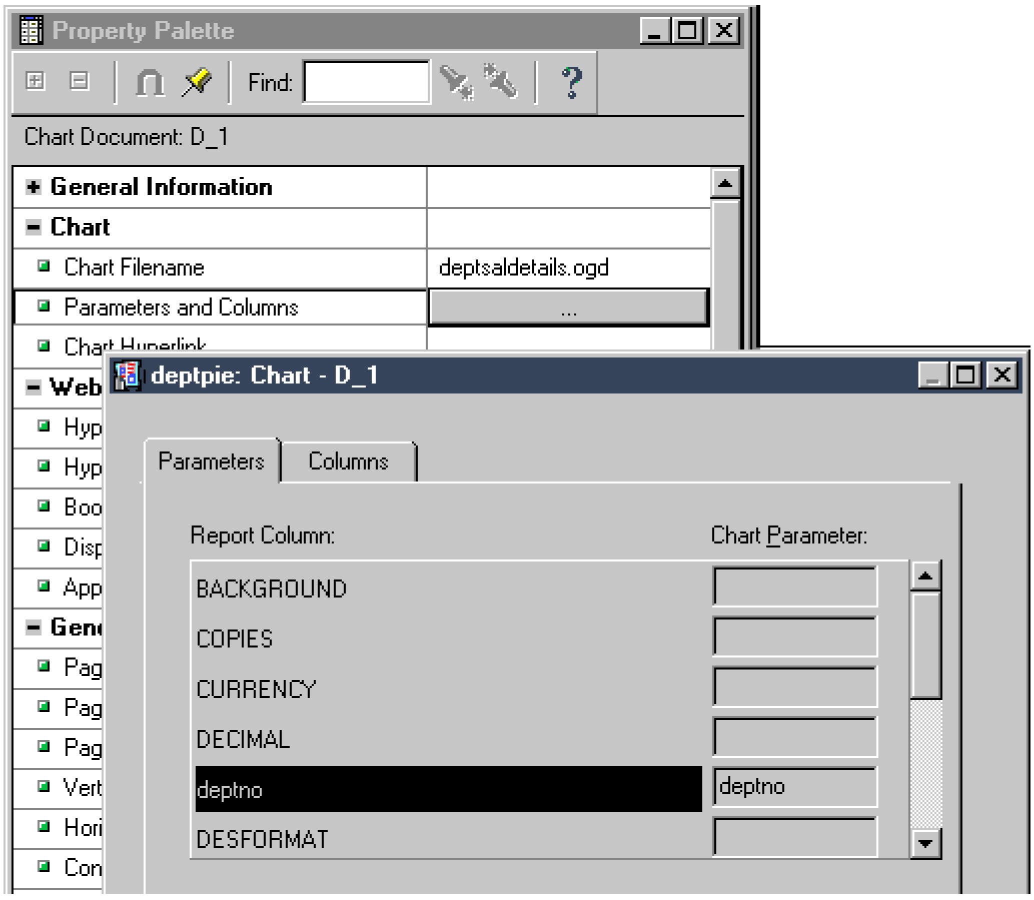


Chart Document: D_1

General Information	
Chart	
Chart Filename	deptsaldetails.ogd
Parameters and Columns	...
Chart Hyperlink	
Web	
Hyp	
Hyp	
Boo	
Disp	
App	
Gen	
Pag	
Pag	
Pag	
Vert	
Hori	
Con	

Report Column:	Chart Parameter:
BACKGROUND	
COPIES	
CURRENCY	
DECIMAL	
deptno	deptno
DESFORMAT	

Figure 4. Setting chart item parameters

3. Move the chart query into the report

In reports the query of the master chart must be executed (though there is no need for this data to be displayed when the report is run, as it is in Figure 2).

1. Open the chart and its query in the Oracle Graphics Builder.
2. Copy the query text.
3. In Reports select *Data Model* from the Object Navigator.
4. Create a *SQL Query* item and paste in the Graphics query text.
5. Re-open the Property Palette of the Reports chart item, and select *Parameters and Columns.*
6. In the resulting dialog box, select the *Columns* tab.
7. In *Chart Query*, enter the name of the query as it appears in the OGD (e.g., “query0”).

8. From the *Report Group* pulldown list, select the reports group that corresponds to the *SQL Query* you have just created in Reports.

9. Highlight each of the columns listed under *Reports Column*, and enter the corresponding name under *Chart Column*—if you copied the query straight out of Oracle Graphics, the corresponding column names will be identical. For example, *sum_sal* in report group *G_deptno* maps to the column alias *sum_sal* in the chart query *query0*:

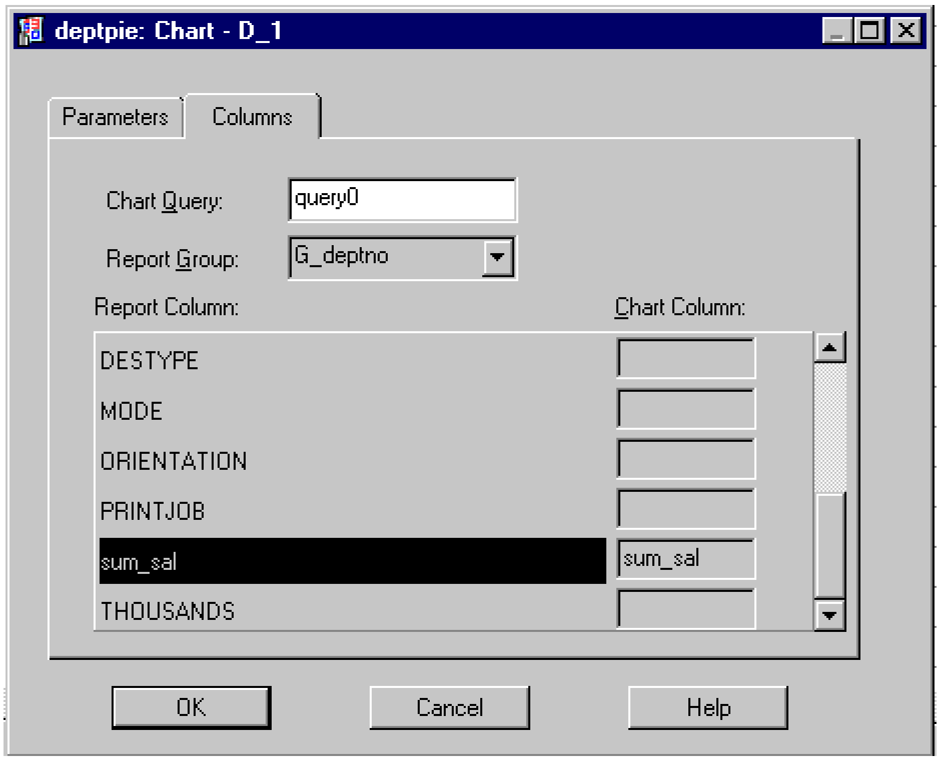

deptpie: Chart - D_1

Chart Query: query0

Report Group: G_deptno

Report Column:	Chart Column:
DESTYPE	
MODE	
ORIENTATION	
PRINTJOB	
sum_sal	sum_sal
THOUSANDS	

Figure 5. Setting chart item column names

Note: If the report column is not highlighted, the correspondence is not registered and an OG-04002 Column not found in query data *error is likely to arise.*

4. Set the chart hyperlink

1. In the report containing the master chart, return to the Property Palette of the chart item.

2. In the property *Chart Hyperlink* (not to be confused with anything under *Web Settings*), enter the URL of the detail report, passing as a parameter the value which corresponds to the selected pie slice, in our example this is deptno. Assuming the two reports are running under the same Reports Server, a relative URL would be enough. So the chart hyperlink might be:

```
/dev60cgi/rwcgi60.exe?report=deptsaldetails.rdf+server=repserve
r6i+DESTYPE=CACHE+DESFORMAT=HTML+deptno=&deptno
```

Notice that userid is not specified in the previous URL example: it is not necessary when one report calls another in this way.

WHAT HAPPENS WHEN THE CHARTS ARE CALLED?

Consider a master-detail pair of charts based on the demo tables, emp and dept, which have been migrated according to these steps. The master, pie chart is based on the query:

```
SELECT d.deptno, sum(e.sal) sum_sal, dname FROM emp e, dept d
WHERE d.deptno = e.deptno GROUP BY d.deptno, d.dname
```

It has the hyperlink:

```
/dev60cgi/rwcgi60.exe?report=enamesindept.rdf+server=repserv6i+
P_DEPTNO=&deptno+P_DNAME=&dname+DESTYPE=CACHE+DESFORMAT=HTML
```

The detail, column chart is based on the query:

```
SELECT ename, sal FROM emp WHERE deptno = :p_deptno
```

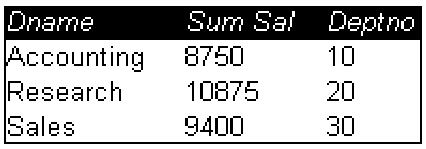
Parameter p_deptno is used to restrict the query, and parameter p_dname is used to format the chart title.

Displaying the master chart

1. The report containing the master chart is called via the Reports Server.

```
/dev60cgi/rwcgi60.exe?report=dnamesal.rdf+userid=sco
tt/tiger@ora816+server=repserv6i+DESTYPE=CACHE
+DESFORMAT=HTML
```

2. The query is executed in the report.



Dname	Sum Sal	Deptno
Accounting	8750	10
Research	10875	20
Sales	9400	30

3. The query data is passed to the chart engine, which in turn creates the chart.

4. The Reports Server displays the chart and retains an image map of the chart linking data items to particular rows in the query.

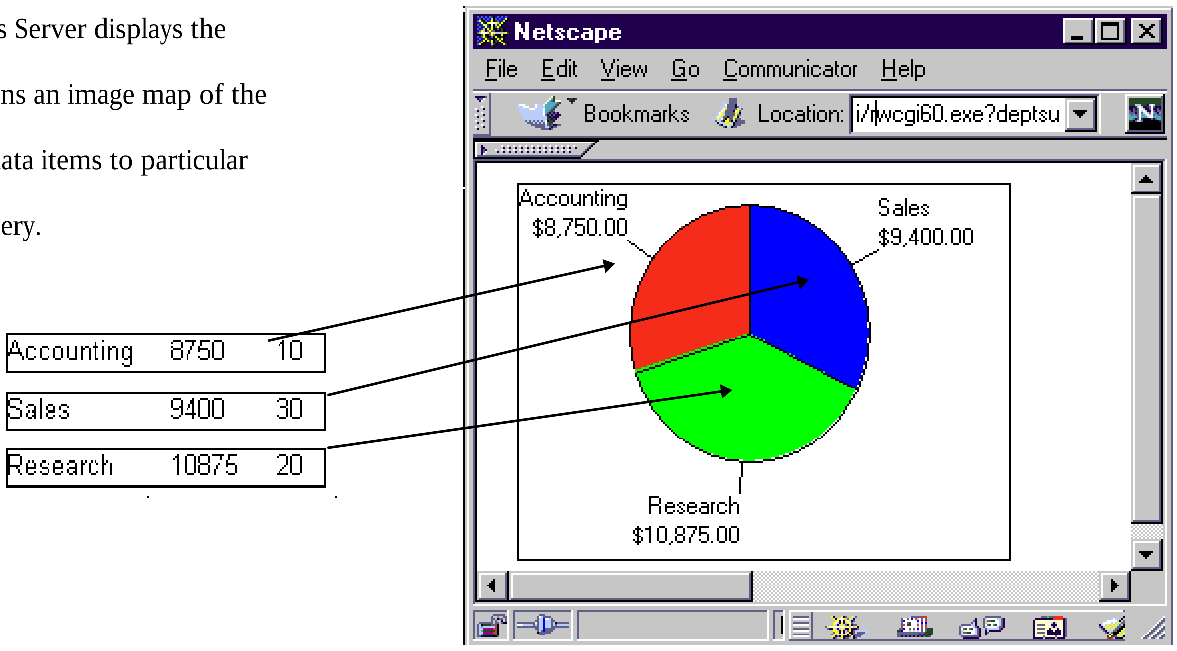


Figure 6. Displaying the master chart

Displaying the detail chart

1. The user clicks on the pie slice for Sales.

2. The Reports Server determines which data row is associated with the pie slice.

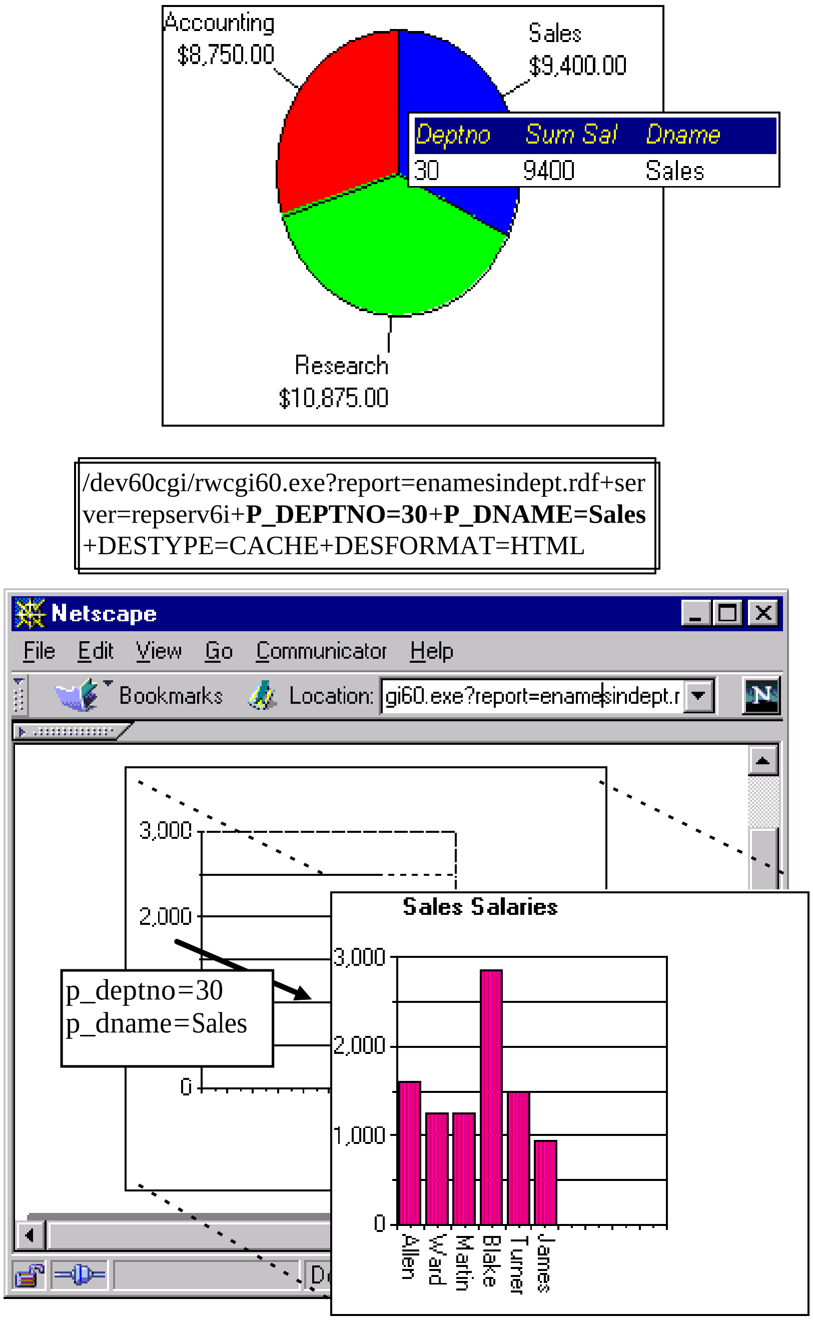
3. Values from the associated data row are substituted for CGI parameters in the chart hyperlink.

4. The report containing the detail chart is called.

5. The report parameters are set.

6. The embedded detail chart is invoked.

7. The chart parameters are set from the reports parameters.



Deptno	Sum Sal	Dname
30	9400	Sales

/dev60cgi/rwcgi60.exe?report=enamesindept.rdf+server=repserv6i+**P_DEPTNO=30+P_DNAME=Sales**+DESTYPE=CACHE+DESFORMAT=HTML

p_deptno=30
p_dname=Sales

Figure 7. Displaying the detail chart

FURTHER CONSIDERATIONS IN USING CHART HYPERLINKS

Two Charts in One Report

In the previous example, we could have moved the query for the detail chart into the report, as we did with the master. We also could have embedded the two charts in a single report, with both their queries executed in the report. The master chart hyperlink would reference the report and pass a parameter that would be used in the query for the detail chart. You may wish to hold off displaying the detail chart until a selection has been made in the master. You could do this using a Reports format trigger on the chart item in the report. Unless the parameter referenced by the detail query is set, the detail chart will not be displayed:

```
FUNCTION DetailChartFormatTrigger RETURN BOOLEAN IS
BEGIN
  IF :PJOB IS NULL
  THEN
     return (FALSE);
  ELSE
     return (TRUE);
  END IF;
END;
```

Referencing a Formula Column in the Chart Hyperlink

A further option is to make the chart hyperlink reference a formula column on the report. This permits more flexible, conditional setting of chart hyperlinks. Depending on the parameter values returned by the chart, completely different URLs could be called. For instance, take a simple report with an embedded pie chart based on the query:

```
SELECT deptno, sum(sal) FROM emp GROUP BY deptno
```

Clicking on the different deptno slices can take the user to completely different Web locations. The formula column would be added via the report's data model: select *Formula column* from the data model tool palette and add it to the group. The data model would look something like this (Figure 8):

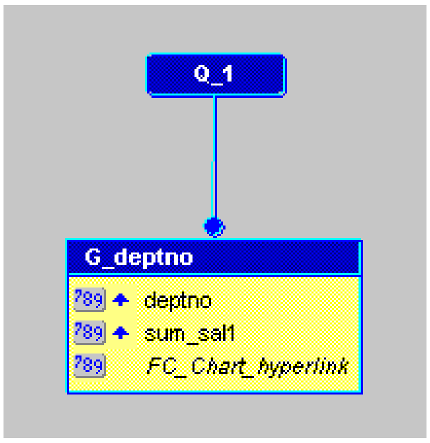


Figure 8. Reports Data Model with Formula Column

Bring up the Property Palette on the formula column and select *PL/SQL formula.* Here we could enter any business logic we like depending on deptno or sum(sal), so long as a valid URL is returned, for example:

```
FUNCTION FC_Chart_hyperlinkFormula RETURN CHAR IS
  hyperlink VARCHAR2(100);
BEGIN
  IF :deptno = 30 THEN
  -- Go to Deparment 30's home page
    hyperlink:='http://myserv.mycomp.com/Research/index.html';
  ELSE
  -- Run the standard departmental report
    hyperlink:='/cgibin/rwcgi60.exe?deptrep+department='
               ||:deptno;
  END IF;
  RETURN(hyperlink);
END;
```

This formula column would then be referenced in the chart item's chart hyperlink property as &<FC_Chart_hyperlinkFormula>. At runtime, selecting a pie slice on the chart causes the execution of FC_Chart_hyperlinkFormula and navigation to the URL which the function returns.

Referencing Frames in Chart Hyperlinks

The Reports 6*i* Demos include an example of a chart hyperlink that references a formula column, repchart1.rdf. The full chart hyperlink on this report is:

```
&<Chart_Element_Hyperlink>" target="reportdemo
```

The reference to the formula column falls between the angle brackets. The rest of the hyperlink concerns the frame into which the URL is to be returned. Double quotes are automatically added at the beginning and end of the chart hyperlink, but if a frame is being targeted, then additional double quotes are required: the first to terminate the URL, the second to mark the beginning of the frame name. At runtime the chart hyperlink might translate as follows:

```
"/dev60cgi/rwcgi60.exe?chart2+PCITY='Boston'" target="reportdemo"
```

The Limitations of Chart Hyperlinks

There are two limitations to using chart hyperlinks for embedding interactive reports:

- Only one report query can be passed to any one embedded Oracle Graphics display.
- Only chart objects that represent data values, such as bars, columns, or pie slices, can be interactive.

If several interactive charts must be retained in a single OGD, or if an OGD contains interactive items which do not represent data values, such as individual button objects, then the built-in OG_SET_URL should be used.

OG_SET_URL AND OG_GET_URL

The OG_SET_URL built-in allows a dynamic URL to be associated with any Oracle Graphics object when it is run using the Reports Server:

```
PROCEDURE og_set_url(any_og_object IN OG_OBJECT,
                     url_text IN VARCHAR2)
```

This URL setting can also be queried using the companion built -in, OG_GET_URL:

```
FUNCTION og_get_url(any_og_object IN OG_OBJECT)
                    RETURNS VARCHAR2
```

Given any Oracle Graphics object – a pie slice, an axis label, a rounded rectangle, and the like – these functions can set and return the URL that is associated with the object. A format trigger is the appropriate place to make this association for an item that represents data – such as a bar, a column, or a pie slice. For items which lack format triggers, this could be done in the OGD's open trigger.

The OG_SET_URL built-in replaces the construction OG_SET_PARAM('OG_URL', *og_url_target*), which was a way of setting a hyperlink on an Oracle Graphics object displayed using the Graphics Cartridge. Code that uses the OG_SET_PARAM built-in to set a URL must be changed in OGDs migrated from the Graphics Cartridge to the Reports Server.

There is a bug, 1423238, with OG_SET_URL in Oracle Graphics version 6.0.8.10.1 (Developer 6*i* release 2) that prevents the built-in from being used in this version. It is fixed in version 6.0.8.12.0.

MIGRATING BUTTON PROCEDURES ON NON-DATA OBJECTS TO USE OG_SET_URL

Consider the simple Oracle Graphics display depicted in Figure 9, a chart and three buttons. Depending on which of the three buttons is selected, *Clerks*, *Managers*, or *Sales*, the chart displays salary details for that particular job category, updating the title appropriately. The buttons set a parameter, :p_job, which is referenced in the query:

```
SELECT ename, sal FROM emp WHERE job = :p_job
```

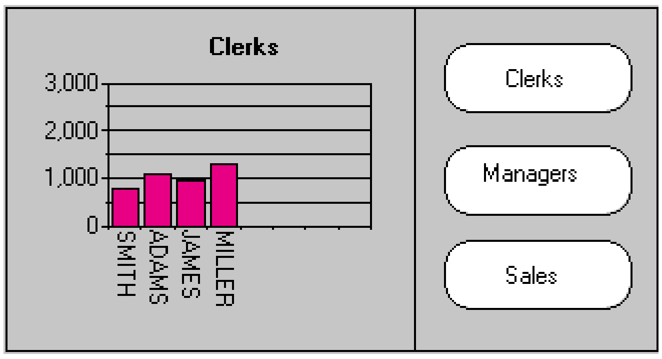


Figure 9. Oracle Graphics display with buttons

The button procedure on *Managers* illustrates how the buttons work:

```
PROCEDURE managerbutton (buttonobj IN og_object,
                        hitobj IN og_object,
                        win IN og_window,
                        eventinfo IN og_event) IS
BEGIN
   :p_job := 'MANAGER';
   OG_EXECUTE_QUERY(OG_GET_QUERY('query0'));
   SetChartTitle(:p_job);
END;
```

This type of button procedure cannot be executed under the Reports Server, but the same functionality can be achieved using the OG_SET_URL built-in.

The modifications required to migrate this chart are as follows:

1. Embed the chart in a report that is run through the Reports Server.
2. Create parameters in the report corresponding to those in the chart.
3. In the report’s chart item Property Palate, under *Parameters and Columns*, in the *Parameters* tab, highlight each report parameter and enter the corresponding chart parameter (see Figure 4, above).
4. In the Oracle Graphics display OpenTrigger, set individual URLs on each of the buttons, for example:

```
PROCEDURE OpenTrigger IS
  vClerkButton OG_OBJECT := OG_GET_OBJECT('ClerksButton');
  vManagerButton OG_OBJECT := OG_GET_OBJECT('ManagersButton');
  vSalesButton OG_OBJECT := OG_GET_OBJECT('SalesButton');
  vReportURL VARCHAR2(200) :=
           '/dev60cgi/rwcgi60.exe?report=jobrates.rdf+'||
           'server=repserver6i+DESTYPE=CACHE+DESFORMAT=HTML';

BEGIN
  OG_SET_URL(vClerkButton, vReportURL||'+p_job=CLERK');
  OG_SET_URL(vManagerButton, vReportURL||'+p_job=MANAGER');
  OG_SET_URL(vSalesButton, vReportURL||'+p_job=SALESMAN');

  SetChartTitle(:P_JOB);
END;
```

When a button is selected, the report will call itself (jobrates.rdf), passing the appropriate parameter value for p_job, which in turn is passed on to the parameter of the same name in the embedded Oracle Graphic display. As with chart hyperlinks, a relative URL for the Reports Server is sufficient, and no connection string is required.

MIGRATING BUTTON PROCEDURES ON DATA OBJECTS TO USE OG_SET_URL

As noted previously, if you want to retain several interactive charts in a single Oracle Graphics display, then you must use the OG_SET_URL built-in instead of a chart hyperlink. This is because only one report query can be passed to any one embedded OGD. On data representing objects, such as a bar, a column, or a pie slice, use OG_SET_URL in a format trigger. In other respects – embedding and passing Oracle Graphics parameters via Reports parameters – the migration is no different.

In the following example (Figure 10), master and detail are on the same display. Click on a departmental pie slice and the chart is called again, this time with the salary details for that department. The selected pie chart slice is even exploded, just by sending the necessary parameters via the report. The parameters are set in the pie slice format trigger:

```
PROCEDURE SetUrl(elem IN og_object,
                 query IN og_query) IS
   vDeptno VARCHAR2(20);
   vExplodeRow VARCHAR2(20);
BEGIN
   vDeptno := TO_CHAR(OG_GET_NUMCELL(query,'deptno'));
   vExplodeRow := TO_CHAR(OG_GET_ROW(elem));
   OG_SET_URL(elem,:ReportUrl||'+deptno='||vDeptno||
'+exploderow='||vExplodeRow);
END;
```

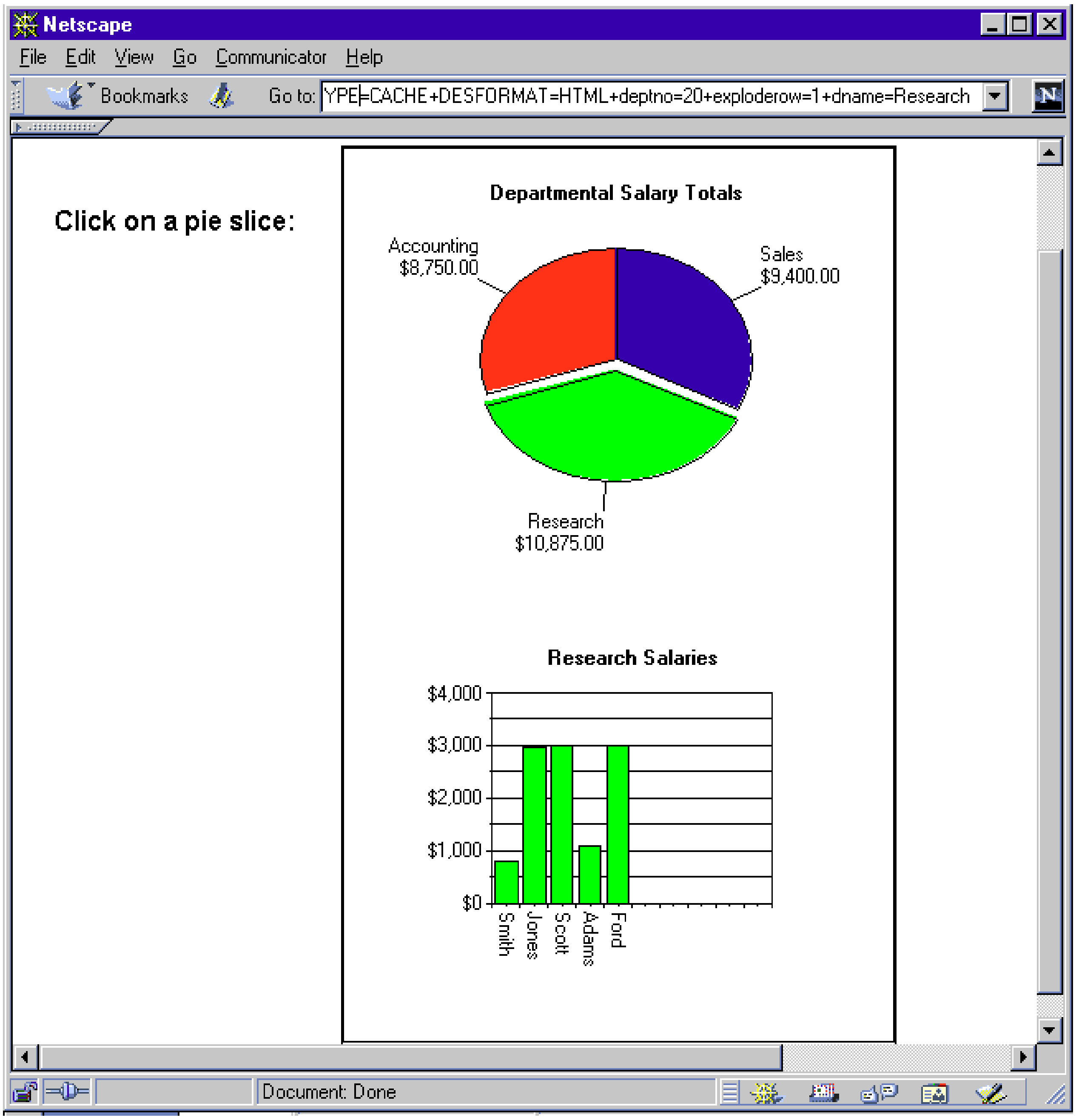


Figure 10. Interactive master and detail charts on the same display

These parameters are then picked up and acted upon in the open trigger and in queries of the chart when it is called again:

```
PROCEDURE OpenTrigger IS
   vChart OG_OBJECT;
   vQuery OG_QUERY;
   vExplodeAmount NUMBER := 10;
   vRow NUMBER;
   vTitle OG_OBJECT;
BEGIN
   /* Disable automatic query execution on the pie chart
   ** so that it can be controlled from the open trigger
   */
   vChart := OG_GET_OBJECT('piechart');
   vQuery := OG_GET_QUERY(vChart);
   OG_SET_EXECOPEN(vQuery, FALSE);
   OG_SET_AUTOUPDATE(vChart, FALSE);
   OG_EXECUTE_QUERY(vQuery);
   /* See if EXPLODROW parameter is not still default of
   ** -1, if not explode appropriately and update pie chart
   */
   vRow := TO_NUMBER(:EXPLODEROW);
   IF vRow >= 0 THEN
     OG_SET_EXPLOSION(vChart,vRow,'salaries',
                      vExplodeAmount);
   END IF;
   OG_UPDATE_CHART(vChart, og_all_chupda);
END;
```

SUMMARY

Oracle Graphics displays can be migrated from deployment via the Graphics Cartridge to deployment via the Reports Server without any loss of functionality. The Reports Server should also prove to be a more robust and efficient method of deployment.

FURTHER READING AND REFERENCES

Oracle Reports Developer Release 6i Online Manuals: Publishing Reports

Explains how to set up the Reports Server and the concepts behind it.

Oracle Report Builder Online Help: About chart hyperlinks

The Reports online help provides further information on chart hyperlinks.

Oracle Reports 6i Demos: Embedding Chart Hyperlinks

The Oracle Demo Tables, script demobld.sql

The script to create the emp and dept demonstration tables used in the examples in this paper can be found under the Developer 6*i* Oracle Home in the DBS directory.