

# **Oracle Reports Remote Printing Utility – 1.3.4**

---

**An Oracle Technical White Paper**

**October 2001**

## PURPOSE

This document describes the Oracle Reports Remote Printing utility (Orarrp) and describes how to use it to provide local printing for users of Web-based applications.

## CHANGES IN DIFFERENT VERSIONS OF ORARRP

### RELEASE 1.2

Initial release on OTN. Oracle 9i Application Server contains default MIME type in the 6iserver.conf configuration file.

### RELEASE 1.3.1

Added MIME Types to better support Windows 2000. The MIME types used are

```
AddType application/x-orarrp      rrp_  
AddType application/x-orarrp-text rrp_  
AddType application/x-orarrp-ps    rrp_  
AddType application/x-orarrp-pdf   rrp_  
AddType application/x-orarrp-rtf   rrp_  
AddType application/x-orarrp-ini    rrp_  
AddType application/x-orarrp-html  rrp_
```

These MIME Type changes are not reflected in 9i Application Server 1.0.2.x per default. Release 1.3.2 in backward compatible to work with MIME type definitions of Release 1.2

### RELEASE 1.3.4

In their 2000 release of Winword Microsoft has changed the API for calling the print dialog. This causes some problems when running Reports output in RTF format not printing properly. To solve this problem the mime type for RTF files has been changed to

```
AddType application/x-orarrp-rt  rrp_
```

and the mime type

```
AddType application/x-orarrp      rrp_
```

No longer exists.

Note that with removing the *application/x-orarrp* MIME type, orarrp 1.3.4 is no longer compatible with release 1.2. Using orarrp 1.2 configurations with orarrp 1.3.4 may lead to orarrp not printing the document.

For RTF files handled by orarrp to print correctly using Winword 2000 edit the ini file located in the directory from where you installed orarrp to use the `rtf_shell_print` option.

[advanced options]

`rtf_shell_print = yes`

## MOVING FROM A PREVIOUS RELEASE TO ORARRP 1.3.4

If you don't have any version of orarrp installed then skip this section.

Before installing orarrp 1.3.4 it is recommended to remove old orarrp versions from the client. Uninstalling a previous release of orarrp will clean the Windows Registry from every occurrences of orarrp mime type settings and remove the orarrp ini file.

Type "orarrp-x" on the command line in the directory where orarrp.exe is located. Note that the orarrp 1.3.4 executable can be used to delete old orarrp releases.

## SUPPORT FOR ORARRP

Orarrp is a sample, is supplied "as is", and is not currently a supported product. Users are encouraged to share their experiences with Oracle and other users of the utility on the 'Oracle Reports' discussion forum on OTN (<http://otn.oracle.com>).

If you encounter any problems using orarrp then share this on OTN (<http://otn.oracle.com>) as well. The OTN discussion forum is monitored by Oracle and once in a while we have a look at our utility to fix or work around known problems. This work depends on customer feedback though.

## INTRODUCTION

In a client/server configuration, the Reports background engine runs on the client's desktop and, for printing, accesses a local printer or a network printer known to the client. This is also true for a Forms application calling a Report to print.

When you run a Report on the Web, all the reporting takes place on the middle-tier server. In this configuration, the Report output is displayed in the user's Web browser (or if called from Forms,

then in a browser after being called via *WEB.SHOW\_DOCUMENT*), and to obtain a printout of the Report a user simply selects 'Print' from the browser's menu.

You may, however, wish to print directly to a local printer without first displaying the output in a browser. One way to do this is to configure the middle-tier server that the Reports Server is running on to have access to any printer in the enterprise that it might need to print on. Then, set the Reports *destype* parameter to "printer" and the *desname* parameter to the name of the desired network printer. In this way, output can be delivered directly to the user's printer. While this solution is suitable for small organizations, it may be difficult to administer in enterprises with hundreds of printers.

The Oracle Reports Remote Printing utility provides an alternate solution that bypasses the requirement that the Reports Server machine have access to all of the printers in the enterprise.

## WHAT DOES ORARRP DO?

Orarrp takes the output of a Report run on the middle tier and causes it to be printed "locally"—to a printer you choose. Orarrp can use any printer you can normally use, be it a workgroup printer or a printer directly attached to your machine.

The way Orarrp handles printing depends on the output type provided from the middle tier. If the Report creates Adobe Acrobat (.PDF) output, then Orarrp invokes the print mechanism for Acrobat, which will include all of the print options provided by that application. The same is true if the output is Rich Text Format. In this case, Orarrp uses your word processor print facility, which will include all of the print options it provides. If the middle tier output is plain text, Postscript, or PCL, then you choose the destination printer and Orarrp handles the rest.

## HOW ORARRP WORKS

Orarrp uses the same mime type mechanism that browsers use to identify Plug-Ins and other helper applications that display non-HTML or Text information. For example, most browsers recognize that the "application/pdf" mimetype is handled by the Adobe Acrobat Reader, and launches it.

In the case of Orarrp, the mimetype "application/x-orarrp" is associated with the Orarrp.exe program. To service this mime type, Orarrp.exe is installed on client machines as a Plug-In.

## INSTALLATION OF ORARRP

Installing Orarrp involves work on both the middle tier server and on each of the client machines.

### SERVER CONFIGURATION

On the server side, a special MIME type must be defined in the Web Server Listener configuration file. Different Web servers register MIME types in different ways. For example, for the Oracle HTTP Server *powered by Apache*, the MIME type is set with the following line in the MIME Types section of the httpd.conf file:

```
AddType application/x-orarrp-text      rrpt
AddType application/x-orarrp-ps         rrpp
AddType application/x-orarrp-pdf        rrpa
AddType application/x-orarrp-rt         rrpr
AddType application/x-orarrp-ini        rrpi
AddType application/x-orarrp-html       rrph
```

These settings also have been added in the 6iserver.conf file (which is sourced by oracle\_apache.conf, which in turn, is sourced by httpd.conf) for newer releases (post August 2001) of Oracle 9i Application Server.

**Please** note that above mime types are the result of latest changes in orarrp fixing reported problems. If you are running Oracle 9i Application Server 1.0.2.x the you need to edit the 6iserver.conf file in the “conf” directory of the Oracle Home where Reports and Forms are installed in. Remove the *application/x-orarrp* mime type and add the new mime types as listed above. Please carefully remove all other orarrp mime types before setting the new type definitions.

There are two bits of information here. First, there is the MIME type: application/x-orarrp. Second, there are the file extensions that the Web server will associate with this particular MIME type:

- *rrpt*, denotes PostScript and PCL.
- *rrpp*, denotes PostScript and PCL.
- *rrpa*, denotes Adobe Acrobat.
- *rrpr*, denotes Rich Text Format.
- *rrpt*, denotes pure text files.
- *rrpi* is a special case. It allows you to post updated versions of the ORARRP.INI file on a Web page (see Posting INI File Updates for Download).

These values are all prefixed with “rrp” for Remote Report Printing. You can change this if you already have an application that uses files with these extensions.

Restart your Web listener to activate the new MIME type mappings.

Please edit the Apache httpd.conf file or the 6iserver.conf file respectively, adding this extension if not yet set in your 9iAS installation. Refer to the listener documentation for information on how to do this.

## CLIENT CONFIGURATION

*Note: Orarrp currently works on any Win32 platform with both Netscape version 4+ and any version of Microsoft Internet Explorer.*

For each client:

1. Download the orarrp.exe file to each client machine.

One way to do this is to place a link to the orarrp.exe file on a Web page. Users can click the link to download and save the exe file to their local disks.

2. Save the orarrp.exe file in the folder c:\program files\orarrp.
3. Double-click orarrp.exe within Windows Explorer. And Click OK when asked if you want to install.

This will create an ORARRP.INI file and set up the local MIME type mappings that the browser will need to invoke Orarrp successfully. Orarrp displays a dialog confirming a successful installation.

4. Restart each client's Web browser.

The first time you use Orarrp, you might be asked if you want to “Open the File from its current location” or “Save this file to disk.” Select “Open the File,” and un-check the “Always ask before opening this type of file” check box.

## POSTING INI FILE UPDATES FOR DOWNLOAD

In most circumstances, the default settings used by Orarrp should be suitable for most users. However, if required, an altered set can be posted on a Web page for users to apply to their local configuration.

To carry out this action the administrator needs to do the following:

- 1) Create an appropriate INI file with all of the required settings

- 2) Change the extension of this INI file to “rrpi”.
- 3) Add a link to the rrpi file from a suitable Web page.

The new INI settings then need to be applied, either by the user explicitly clicking on a link to the rrpi file, or by some automated means, for instance, a WEB.SHOW\_DOCUMENT() call from within a Forms Server application.

## TESTING ORARRP

Once you have configured the server and the client, test Orarrp to ensure that the MIME types are correctly set up.

A simple test is to take an existing Adobe Acrobat file and change its file extension from **.pdf** to **rrpa**. Then create a simple Web page on your Web server that points to this renamed document. For example, we'll assume that your renamed pdf file is now called test.rrpa. Create the following simple Web page:

```
<html>
<body>
  Orarrp Test; Click here to Print:
  <a href="test.rrpa">Acrobat Doc</a>
</body>
</html>
```

Clicking the link should download the file to the client and open the Acrobat Printer dialog.

## USING ORARRP

The examples in this paper address situations where Oracle Forms and Oracle Reports are used together; however, you can also use Orarrp to print Reports output locally, for example, from an Oracle Portal page or a plain HTML page.

Assuming the Forms/Reports scenario, here is the sequence of events that occur when Orarrp is handling printing:

1. The Forms application calls a Report and specifies that it should be output to file with the applicable Orarrp extension (see below).
2. The Report runs and places the output on the Web server where it will be visible to the user.

3. The Form monitors the progress of the Report, and, when it is complete, issues a `WEB.SHOW_DOCUMENT()` call to direct the user's browser to the generated output file.
4. The Web server recognizes the output file's extension as being of MIME type "application/x-orarrp" and streams the data to the browser with that MIME type.
5. The browser recognizes the "application/x-orarrp" mime type and invokes the `orarrp.exe` program to handle it.
6. `Orarrp.exe` detects the type of output (e.g., Postscript/PCL, Acrobat, etc.) and either invokes the correct helper application to print, or displays its own printer selection dialog to the user.
7. The document prints!

## ORARRP FILE EXTENSIONS

For `Orarrp` to work correctly, the output from Reports must be saved to a file with the correct file extension. `Orarrp.exe` uses the file extension on the browser client to determine how to print the file. The following table lists the types of output that Reports can generate and the extension to use to name the output for `Orarrp`.

Format	File Extension	Notes
PostScript	rrpp	Printed directly by <code>orarrp.exe</code>
PCL	rrpp	Printed directly by <code>orarrp.exe</code>
DELIMITED	rrpt	Printed directly by <code>orarrp.exe</code>
PDF	rrpa	Passed to Acrobat to print
RTF	rrpr	Passed to word processor to print  Note: For RTF files handled to print correctly using Winword 2000, edit the ini file located in the directory from where you installed <code>orarrp</code> to use the <code>rtf_shell_print</code> option.  [advanced options]  <code>rtf_shell_print = yes</code>
Plain Text	rrpt	Printed directly by <code>orarrp.exe</code>



INI File Update	rrpi	This file is not printed, but the contents will be applied to the user's ORARRP.INI file. This affords a way for administrators to provide a standard INI file for all users.
HTML, HTMLCSS, HTMLCSSIE	rrph	Printed directly by orarrp.exe. <i>Note: That only the HTML itself and NOT embedded images will print.</i>

*Note: XML format is not supported by Orarrp.*

## AN EXAMPLE OF THE FORMS CODE USED TO CALL ORARRP

The following examples demonstrate the type of code you would use to run a Report and print its output locally via Orarrp.

The Forms Code uses RUN\_REPORT\_OBJECT() to execute the Report. We use this built-in to track the progress of the Report from a timer and to print once the Report has finished.

The code consists of two triggers:

- WHEN-BUTTON-PRESSED—This could be any trigger, but it illustrates the use of RUN\_REPORT\_OBJECT to start the Report, and the creation of a timer to monitor its progress.
- WHEN-TIMER-EXPIRED—We check the Report status on a regular basis using this timer trigger. Once the Report has finished we can print it.

## Launch the Report (When-Button-Pressed trigger)

```

DECLARE
    Report_Id                report_object;
    hTimer                   timer;
    vcFile                   varchar2(255);
    vcResult                 varchar2(30);
BEGIN
    /* Generate a pseudo-unique filename */
    vcFile := get_application_property(USERNAME)||TO_CHAR(SYSDATE,'YYYYMMDDHHMISS');
    /* We will be producing Postscript output in this case so add a .rrpp extension */
    vcFile := vcFile||'.rrpp';

```

```

/* Find an existing Report Object and set parameters */
report_id:=FIND_REPORT_OBJECT('Forms_Report_Node_Name');
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_EXECUTION_MODE,BATCH);
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESTTYPE,FILE);
/* Format to Postscript */
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESFORMAT,'ps');
/*Desname to our generated filename
    Note: <PhysicalDir> refers to a location on the
    Webserver which we read from with SHOW_DOCUMENT() */
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_DESNAME,'<PhysicalDir>'||vcFile);
/*Set the name of the Reports Server, again this would be softcoded */
SET_REPORT_OBJECT_PROPERTY(report_id,REPORT_SERVER,'Reports_Server_TNS');
/* Save the Report Handle to a global as well */
:GLOBAL.ReportHandle := RUN_REPORT_OBJECT(report_id);

/* Save the generated Filename away for use in the When-Timer-Expired */
:GLOBAL.PrintOutPut := vcFile;
/* Create a timer to monitor the Report,
    This one is set to check 4 times a minute */
hTimer := CREATE_TIMER('PRINTER_QUEUE','15000',REPEAT);
END;

```

## Check the Status of the Report (When-Timer-Expired trigger)

Every time the timer defined in the previous trigger expires, the application will look for the timer's name and get the status of the Report. When the Reports Server indicates that the Report is complete, we can invoke Orarrp to actually carry out the print.

```

DECLARE
    vcStatus          varchar2(50);
    vcTimerName       varchar2(30);
    hTimer            timer;
BEGIN
    vcTimerName      := get_application_property(timer_name);
    IF      vcTimerName = 'PRINTER_QUEUE' THEN
        vcStatus := REPORT_OBJECT_STATUS(:GLOBAL.ReportHandle);
        IF vcStatus in ('RUNNING','OPENING_REPORT','ENQUEUED') THEN
            NULL;
        ELSIF  vcStatus = 'FINISHED' THEN
            /* Point the user's browser at the generated file */

```

```

        WEB.SHOW_DOCUMENT('<VirtualDir>'||:GLOBAL.PrintOutPut,'_blank');
    /* Clean up the timer */
    delete_timer(vcTimerName);
ELSE
    /* The Report has not worked for some reason
       Inform the user */
    MESSAGE('Report Has failed');
    delete_timer(vcTimerName);
END IF;
END IF;
END;

```

## ORARRP TECHNICAL INFORMATION

### CONFIGURATION FILE

Orarrp supports a limited amount of end-user configuration via the orarrp.ini file. This file is automatically created by the Orarrp executable and is stored in the same directory. You can edit this configuration file in a text editor or use a .rrpi file to download new settings. You can add or change the following parameters, if required:

INI File Section	Parameter	Values	Notes
[Install]	extension_root	rrp	If you want Orarrp to use extensions other than .rrpi, rrpt, .rrpa, .rrpr, .rrph, and .rrpp, then you should manually create and configure the orarrp.ini file before you install the executable. Set this parameter to the new extension you want to use. For example, setting the parameter to “xxx” would create associations for Orarrp of .xxxt, .xxxa, and so on.

			<p>Extension_root is used only during install and de-install of the utility.</p> <p>If you change this parameter, then you must also make a matching change in the MIME types setup in your Web server.</p>
[Install]	silent_refresh	yes   <u>no</u>	<p>Indicates if an INI file update via the rрпи mechanism pops up a dialog on completion. If you are automatically refreshing the users INI file, on a regular basis, you will probably want to set this to “yes” in which case the dialog will not be show,</p>
[Options]	choose_printer	<u>yes</u>   no	<p>If set to “no,” Orarrp will send PDF, RTF, Text, PCL, and PostScript output direct to your default printer, without showing a printer selection dialog.</p> <p>If set to “yes,” Orarrp will provide a print dialog where you can choose a printer on-the-fly.</p> <p>Choose Printer has no effect when printing a Rich Text Format document on a machine without Microsoft Word installed.</p>
[Options]	send_form_feed	yes   <u>no</u>	<p>Allows you to add a blank trailing page to PCL and Postscript Documents.</p>
[Advanced Options]	pdf_shell_print	yes   <u>no</u>	<p>If this Option is set to “yes,” Orarrp uses the Windows Shell commands to print the document. This is the equivalent of right mouse clicking on the document and choosing “Print”. Acrobat Reader reacts differently to this command depending on the version installed. Acrobat 4.0 will simply</p>

			<p>print immediately to your default printer, whereas Acrobat 3.0 pops up a printer selection dialog.</p> <p>Setting this value to “no” (the default) allows Orarrp to print the document using the Acrobat programmatic API. We recommend that you keep this setting at “no” unless you are experiencing problems when printing Acrobat documents.</p> <p><i>See the Restrictions section for further information on this option</i></p>
[Advanced Options]	rtf_shell_print	yes   <u>no</u>	<p>If this Option is set to “yes,” Orarrp uses the Windows Shell commands to print the document. This is the equivalent of right mouse clicking on the document and choosing “Print”.</p> <p>Additionally if you don’t have Microsoft Word installed, then this has the same effect as having rtf_shell_print=yes.</p> <p>Setting this value to “no” (as long as Microsoft Word is installed) will use Word’s programmatic APIs for printing. We recommend that you keep this setting at “no” unless you are experiencing problems when printing RTF documents..</p> <p><i>See the Restrictions section for further information on this option</i></p>
[Advanced Options]	postscript_dos_print	yes   <u>no</u>	<p>If set to “yes,” Orarrp will hand the printing of text, PCL, and Postscript documents off to DOS. Switch this on only if you are</p>

			experiencing errors with the default mechanism.
[Advanced Options]	debug	yes   <u>no</u>	If set to “yes,” Orarrp will write debug information to <i>orarrp.dbg</i> , a file located in the same directory as the .ini file and executable.
[Advanced Options]	lock_ini	yes   <u>no</u>	If set to yes, Orarrp will not delete the existing ini file when un-installing, nor will it overwrite it when installing.

*Note: No other parameters in the orarrp.ini file should be changed.*

## DE-INSTALLING ORARRP.EXE

You can remove the entries made by Orarrp in the client machine registry by issuing the command “orarrp -x”. This command cleans up the users registry and ini file only. You must manually delete the orarrp.exe files. If the lock\_ini file option is set to yes in the ini file, that file will not be deleted.

## COMMAND LINE OPTIONS

Orarrp supports the following command line switches:

- i** Install the utility
- x** De-install the utility’s registry entries and ini file
- s** Silent mode. Install or de-install will not display a completion dialog. Errors during the install or de-install process will still be shown in dialogs.
- l** Write debug information to the orarrp.dbg file in the same directory as the orarrp.exe file. This is an alternative to using the debug= switch in the .ini file.
- ?** Display the list of command line options

## ORARRP REQUIREMENTS AND RESTRICTIONS

- Orarrp is a Win32-only utility.
- Orarrp is not NLS compliant and is English only.
- When installing or de-installing Orarrp the current Windows user must have privileges to create, delete, and set registry keys in HKEY\_CLASSES\_ROOT.
- When using Orarrp, the user must have privileges to write to the orarrp.ini file.
- When using Orarrp, the user must have privileges to read from registry keys in HKEY\_CLASSES\_ROOT.
- If PostScript or PCL output is created and sent to the local printer, that printer must be able to handle such output. Orarrp does not interpret the print data at all; it simply spools it to the printer.
- If printing Adobe Acrobat or Rich Text Format, the local machine must have a suitable application registered to handle files of type .PDF and .RTF respectively.
- When printing Adobe Acrobat (PDF) files. If the configuration setting **pdf\_shell\_print** is set to “yes”, then we cannot guarantee that the user will or will not be prompted to select a printer, as this varies between versions of Acrobat Reader. We recommend that the pdf\_shell\_print setting be left at “no”.
- When printing Rich Text Format (RTF) files on a machine without Microsoft Word installed, or where the configuration setting **rtf\_shell\_print** is set to “yes”, we cannot guarantee that a printer selection dialog will or will not appear. This depends on the default print action of the currently installed handler for .RTF files.
- If you are creating PCL or Postscript output for local printing, then the middle-tier machine that is hosting the Reports Server must have access to a PCL or Postscript printer.



Oracle Corporation  
World Headquarters  
500 Oracle Parkway  
Redwood Shores, CA 94065  
U.S.A.

Worldwide Inquiries:  
+ 1.650.506.7000  
Fax + 1.650.506.7200  
<http://www.oracle.com/>

Title: Oracle Reports Remote Printing Utility  
Version: 1.2.2  
Date: December 2000, August 2001, October 2001  
Authors: Duncan Mills, Frank Nimphius  
Editors: Joan Carter  
Contributors: Paul Narth

Copyright © Oracle Corporation 2001  
All Rights Reserved

This document is provided for informational purposes only, and the information herein is subject to change without notice. Please Report any errors herein to Oracle Corporation. Oracle Corporation does not provide any warranties covering and specifically disclaims any liability in connection with this document.

Oracle is a registered trademark, and Oracle8i, Oracle8, PL/SQL, Oracle Reports, and Oracle Forms are trademarks of Oracle Corporation. All other company and product names mentioned are used for identification purposes only and may be trademarks of their respective owners.

---