Using Radar for Ultra-Scalable File Detection and Server Process Execution

Using Radar with Sterling Connect:Direct for z/OS 5.1.0 Reference Implementation

Copyright © 1999, 2005, IBM Sterling Corporation. All Rights Reserved.
Last Updated: 11 Nov 2005 (Last updated 11 Nov 2005)

Contents

Introduction
Hypothetical Business Requirements
Solution Overview
Configuring a Radar Gateway to Spawn an Instance of Uploader
Turning On the Scanning Engine
Troubleshooting
Troubleshooting
Conclusion

Introduction

This document provides an example of a highly scalable way in which you can use Radar to monitor a Sterling Connect:Direct for z/OS (formerly Sterling Connect:Direct for OS/390) node for new files and Sybase/Open Client and Sybase/Open Client/C-Gateway for z/OS for stored procedures. The example follows the flow of file events in a business partner data flow.

The document is intended to serve as a reference implementation for software engineers and IT decision makers who are considering or beginning to deploy Sterling Connect:Direct and Sterling Radar.

Before delving into the configuration steps of this reference implementation, a brief background on what Radar is and how it can be used to solve a business need is in order. If you are familiar with the Tail, a high level overview of Radar, click here to skip the background information and go to the Hypothetical Business Requirements section.

Hypothetical Business Requirements

A growing company called "YouCompany" that is receiving more orders than it can handle has a business need. It has decided that it wants to automate a portion of its order fulfillment process by sending orders from their ERP system to a third party fulfillment vendor called "ThirdParty".

Solution Overview

"YouCompany" decides that FTPing an ERP system extract of the order data to "ThirdParty" is a cost effective solution. Since FTPing is a common, but much more the importantness, providing that there is some way to guarantee the reliability of the FTP process, it is a good candidate for a reference implementation that follows. Because "YouCompany" also uses Sybase/Open Client and Sybase/Open Client/C-Gateway, a stored procedure execution is needed to implement a guaranteed delivery FTP service.

Configuring a Radar Gateway to Spawn an Instance of Uploader

NOTE: If you will be doing along with the implementation steps in a testing mode on your own computer, you can download and install a free trial of the Sterling Connect:Direct Secure+ Client.

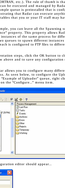
The basic configuration with for Radar is a gateway. Radar spawns out a file difference that a traditional message queue type of queue. Radar spawns out a message that contains the name of the file that changed. The Radar Gateway that appears in the Tables that the queue is mapped to. Radar will spawn an associated process. It is sent to the screen shot below, the relationship between Tables, Queues and Gateways is shown in the Radar Gateway Manager.

The following queue properties editor will appear.

As you can see above, there are many properties that can be set for a Radar Gateway. Because this document is a reference implementation, for the purposes of this example, you can leave all the values at their defaults. You can should set the "Spawnmax scan frequency" to 1 and set the scale to "Seconds" if you want to see the results of your work in a reasonable time. Leave the scan frequency at 60 seconds if you want to see the results of your work in a reasonable time.

All the properties are fully documented in the help files but it is worth mentioning a few of the more about some of them here before proceeding. The "Spawnmax" properties are used to limit the number of processes that can be spawned the Radar scanning engine will be looking for files to appear. You can set "Spawnmax" to 0 if you want to. The "Spawnmax" properties are used to limit, well, a shared resource, the "Spawnmax scan frequency" changes allows you to adjust the scan rate (i.e. Now often the Tables will be scanned).

Click on the "Spawnmax" properties tabbed on the left hand side of the box shown above and another group of properties will appear as seen below.



The Spawning properties allow an executable to be spawned with the queue. In this case, the executable associated with the queue is the Sybase/Open Client/C-Gateway for z/OS. Radar can be configured to spawn a Sybase stored procedure. Radar can be configured to spawn a Sybase stored procedure with the name. In the same way, Radar is responsible for spawning files via FTP, Radar can be and usual Radar can be configured to spawn an executable that will perform FTP.

Standard
Process Description

Uploader Imports data into ODBC compliant database
Sybase Imports data into Sybase/Open Client/C-Gateway
Uploader Uploads files via FTP
Sybase Runs Sybase/Open Client/C-Gateway stored procedures via RPC
Radar Runs files to computers via the ODBC compliant data
Radar Delivers email messages and psychotronics via SMTP
Sybase Runs Sybase/Open Client/C-Gateway
Sybase/Radar Runs of the execution of scripts. REXX/Perl

More detailed information on the role of each process can be obtained in the SybOpen/C-Gateway Integration Guide (2011.7.10) white paper.

In addition to the executables listed above, Radar can be configured to spawn any other executable. Radar can be configured to spawn a Perl, REXX, Java, COBOL, Microsoft .Net, or C/C++ executable. Radar can be configured to spawn a Perl, REXX, Java, COBOL, Microsoft .Net, or C/C++ executable and managed by Radar. To configure Radar to spawn a Perl, REXX, Java, COBOL, Microsoft .Net, or C/C++ process Manager. The documentation for Radar contains examples of how to do this. The Sybase/Open Client/C-Gateway for z/OS is a common and strongly utilized.

For the purposes of the example, you can leave all the Spawning settings as they are. Note the "Interval" property. This property allows Radar to be configured to spawn a process at a set interval. This is useful for processes that need to be run on a set time or spawn different iterations of the same process (for example, a process that FTPs files to different FTP servers).

Continuing with the implementation steps, click the OK button to close the Queue properties editor and click on the "Gateways" tabbed on the left hand side that one may have made.

As mentioned in step 1, Radar allows you to configure many different iterations of the same process. In case below, to configure the Uploader process, click on the "Add" button. The following Gateway editor will appear. Fill in the Uploader name and click on the "Configure" menu item.

The following Uploader configuration editor should appear...

5. Set the Uploader process's exec fields. For the purposes of this example, default alternatives will be selected to intentionally cause a failure.

6. Click the "Exclusive exec logging" option.

IMPORTANT: For the purposes of this example, this option must be checked in order to see an important logging configuration issue.

7. Click the OK button to close each Uploader property editor and save any changes that you may have made.

All necessary configuration tasks are now complete. The next section will show how to turn on the scanning engine to begin processing files.

Turning On the Scanning Engine

With the configuration complete, the Radar scanning engine can now be started. To turn the scanning engine on, click the OK button button on the toolbar seen below. Alternatively, click on the "Start/Stop" menu item as shown below.

Within 5 seconds, Radar should detect the file that you placed in the queue directory. Because the Uploader process is set to fail, the Radar Gateway will be set operational within 5 seconds. Click on the "Refresh" button to see the Radar Gateway's status.

For the purposes of this example, the instance of Uploader, if configured as it was above, will experience a failure and the process will exit with 12 (non-zero failure) going up. A success, like the one below, will appear when Uploader is interpreting a successful.

To continue on with the example and to highlight some of the important configuration issues, click on the "Local" button. When presented with a cancel confirmation prompt, click the "Yes" button.

Radar will now be operational but minimally configured to fail, so we proceed with some troubleshooting.

Troubleshooting

Below are the basic troubleshooting steps that can be taken to diagnose the Uploader configuration.

1. The first place to look for information about a process failure is the "Last Run" field associated with the process resource. For example, below is how one can see what happened during the execution of the Uploader process.

2. The next logs fired every time a process takes during execution. In case above, when a task is successful, there will be a green check in the left of it, when it fails there will be a red dot in it. Below shows that an error log is fired when the Uploader process failed. To see the error log, double click on the error log by clicking on the "Errors" tab on the left hand side.

3. Open spawnd, type Monitor (or and a log entry should appear as seen below.

4. As seen below, double click on the log entry and look in the lower pane to examine the log and determine the nature of the failure.

In the case of the example, examination of the logs makes it clear that the FTP job failed because "uscomparagraphser.com" is not a valid host.

Once one has determined the root cause of the failure, recovery is made very simple thanks to the advanced power pending file testing and management capabilities of Radar. Because Radar is a file difference product, it will continue to monitor the work space directory for a file that matches the configuration specified in the Queue "Tables" linked associated with the queue when the Uploader process failed at 12:00 for the

So now above, to restart the Uploader process in FTP file for, all you have to do is select the file and click on the repeat button and Radar will automatically re-spawn the Uploader the queue for detection and processing.

Conclusion

The example described in the reference implementation is one of many possible implementations. Radar is a very powerful product and can be used to provide a valuable addition to any integration solution. If you have not already done so, please consider downloading a free trial of Sterling Radar. Also, you can get yourself by downloading a free evaluation copy of the Enterprise Integration Suite.