Simulation Driver and Radar Recorder (SDRR)

User Reference Guide

(JVN – UG – 001)

March 31, 2019

Volume 4: User Reference Guide SDRR v10.4.6

JVN Communications Inc.
2511 Fire Rd. Suite A4
Egg Harbor Township, NJ 08234
609-569-1477

TABLE OF CONTENTS

1. SDRR Overview

The Simulation Driver and Radar Recorder (SDRR) is a simulation injection tool that can be configured to run in various environments. It allows injection of local, interfacility, and surveillance messages, and provides a display of messages exchanged with all configured facilities and devices. The display provides message counts, device status, and error messages.

SDRR can be used for injection of exported scenarios where flight data messages and surveillance data have been previously generated and written to files. This allows SDRR to be used for testing where inputs are consistent and repeatable from run to run. Alternately, SDRR can be configured to receive flight data and target positions dynamically from Dynamic Simulation (DYSIM) instead of from exported scenario files. This allows for interaction with the simulation by Air Traffic controllers and pilots and coordination with strategic systems like TBFM.

2. Getting Started

2.1. Environment Variables

SDRR uses several environment variables to set the locations of configuration files, scenarios, and logs.

Table 1. Environment Variables

Variable	Description
SDRR_CONFIG_PATH	Location of SDRR configuration files (e.g., /usr/local/cfg).
SDRR_SCENARIO_PATH	Location of SDRR scenario directories (e.g., /usr/local/scenarios).
SDRR_LOG_PATH	Location of SDRR log files (e.g., /usr/local/log).
RECORD_PATH	Location of Recording files (e.g., /usr/local/recordings).

2.2. SDRR Parameters

SDRR can be started with various options which control its operation.

Table 2. Program Parameters

Parameter	Description
<sdrr_config.xml>	The SDRR configuration xml file argument is required to start the SDRR software. A relative path of /usr/local/cfg/ is assumed, unless a full path beginning with "/" is used.
-s <sdrr_scenario.xml>	Start SDRR with a an exported SDRR scenario.
--cmdDev=tcps:<sdrr_address>/<port#>	Directs SDRR to listen for simulated flight data messages from DYSIM over a TCP connection on the named address and port. The DYSIM executable, such as simDriver, must be started with the corresponding parameter.
--dysimDev=tcps:<sdrr_address>/<port#>	Directs SDRR to listen for simulated target position data from DYSIM over a TCP connection on the named address & port. The DYSIM executable, such as simDriver, must be started with the corresponding parameter.
--start	Commands SDRR to start running as soon as it comes up.
--utc	Use system time for scenario messages, not the scripted scenario timestamps.
-n, --noscenario	Start SDRR with a configuration file only, without loading an exported SDRR scenario. Injections will come from DYSIM.
-f, --nofullscreen	Not full screen mode.
-v, --version	Displays SDRR version.
-h, --help	Display application parameters.

2.3. Starting SDRR

2.3.1. Starting SDRR with the Startup Wizard

The SDRR Startup Wizard can be launched either by left clicking on the SDRR icon on the lower right side of the display (locally on the server or through a VNC session) or by typing **sdrr** at a command line:

```
> sdrr
```

Once the Startup Wizard is launched, the Select Configuration menu appears. This shows expandable directories in black and selectable configuration files in green. For configuration files to appear selectable green, they need to end in .xml. Users may need to expand the Name bar to the right to see the entire filename. The Comments are displayed from the comments section of the configuration file. User can add or change these comments. These files are located in the /usr/local/cfg directory. Select the Next button to continue to scenario selection.

The Select Scenario menu displays directories and scenarios. This menu is populated from the /usr/local/scenarios directory. Selecting a scenario is optional, you can click next without selecting a scenario to test your configuration file or test Interfacility connections. Click Next to continue.

The Select Options menu displays control options. If a scenario was selected in the previous step, the values displayed will be read from the scenario sdrr.xml file found in the /usr/local/scenarios/<scenario> directory. This file can be modified to reflect some of the default options. Once all selections are made, click Finish to load SDRR. The available options are:

Time

Scenario start time allows for setting the start point in the scenario. The time displayed comes from the **sdrr.xml**.

Date

Scenario date. Useful for multi day scenarios; it can help advancing days.

Controls

Start on GI

Starts upon receiving GI message from HOST. The HOST sends a GI message to inform attached devices, when it's ready. SDRR will start when this message is received.

Autoset Starttime

SDRR will determine start time based on first radar message (time stamp from the .srv files; usually used to playback live radar recordings).

UTC Mode

Synchronizes scenario to system time. It will adjust system message time to UTC format. All messages containing time substitution markers ($....$) will be adjusted.

Auto TA

SDRR will generate TA messages in response to TI messages.

Record Select Indicators

Messages tagged with the selected indicators will be injected.

Log File

Location of the SDRR log file (rename to the scenario related name for easier tracking).

2.3.2. Starting SDRR with Command Line Options

To bypass the Startup Wizard, SDRR can be started from the command line with a configuration file, a scenario, and other optional parameters specified. To launch SDRR, enter:

```
> sdrr <sdrr_config.xml> -s <sdrr_scenario.xml> <optional parameters>
```

To launch a version of SDRR that is not the default version, enter:

```
> /usr/local/jvn.x.x.x/bin/sdrr <sdrr_config.xml> -s <sdrr_scenario.xml> <optional
  parameters>
```

2.3.3. Starting SDRR for Dynamic Simulation

For dynamic simulation, SDRR must be started from the command line in order to set the parameters for message exchange with the DYSIM executable. The DYSIM executable must be started with the corresponding parameters. To launch SDRR, enter:

```
> sdrr <sdrr_config.xml> --cmdDev=tcps:<sdrr_address>/<port#>
  --dysimDev=tcps:<sdrr_address>/<port#> --start --utc -n
```

To launch a version of SDRR that is not the default version, enter:

```
> /usr/local/jvn.x.x.x/bin/sdrr <sdrr_config.xml>
  --cmdDev=tcps:<sdrr_address>/<port#> --dysimDev=tcps:<sdrr_address>/<port#>
  --start --utc -n
```

3. Configuration Files

The SDRR configuration file can include configuration for en route facilities, terminal facilities, and several other types of interfaces. The way these interfaces are defined determines the simulation mode and the types of messages that will be sent by SDRR.

3.1. Simulation Modes

SDRR can be configured in several ways depending of the systems that will be physically connected and which systems will be emulated by SDRR. For simulation with TBFM as the main system under test, SDRR can be configured in direct mode, mixed mode, and indirect mode.

3.1.1. Direct Mode

In direct mode, SDRR is configured to emulate the NAS en route and terminal systems that provide data feeds to TBFM. For the en route data, SDRR generates CMS messages for injections into an EDDS, emulating the feed that would be provided by ERAM. For terminal data, SDRR generates AIG messages that are sent directly to TBFM, emulating the feed provided by STARS.

In order to emulate ERAM, SDRR must be configured with an eramsim source definition and an eddserver definition with connection information for an EDDS server. To connect to an EDDS server, the SDRR configuration file needs to have the "listenAddress" set to a network interface address of the processor running SDRR. On the EDDS server, the configuration file /opt/ssr/hid_address.adp should have the same address for the E1 attribute. Both files should have the same port configured.

In order to emulate STARS and inject AIG messages directly into TBFM, each STARS site should be added to the SDRR configuration file inside the eramsim stanza and also as a stars stanza. The sites should have the facility name as it is adapted in ERAM, and the TBFM name. The multicast addresses and ports defined in the TBFM customization should be added as tsas datasets inside the stars stanzas.

```
<eramsim name="zla">
   <eddserver port="50020"  listenAddress="tbfmsdrr01"/>
   <stars name="ttt" device="tcps:tbfmsdrr01/19000" tangent="+33:47:30.41,-
   118:00:08.06" magdev="14.0"/>
</eramsim>
<stars name="ttt" device="tcp:tbfmsdrr01/19000" facName="TTT" rxclock="0"
autoTR="0" txclock="0">
   <hostio name="zla" facName="zcl" tangent="+33:47:30.41,-118:00:08.06"
   magdev="14.0"/>
   <tsas tbfmName="LAX" potOffset="1154.65,1706.24" pot="+33:58:50.00,-
   116:59:27.00">
```

```
        <dataSet id="7"
        outDev="(multi://224.100.100.107:19000?ttl=3+multi://224.100.101.107:19
        000?ttl=3)"/>
        <dataSet id="8"
        outDev="(multi://224.100.100.108:19000?ttl=3+multi://224.100.101.108:19
        000?ttl=3)"/>
        <dataSet id="9"
        outDev="(multi://224.100.100.109:19000?ttl=3+multi://224.100.101.109:19
        000?ttl=3)"
        inDev="(multi://224.100.100.109:19001+multi://224.100.101.109:19001)"/>
        <dataSet id="10"
        inDev="(multi://224.100.100.110:19001+multi://224.100.101.110:19001)"/>
    </tsas>
```

3.1.2. Mixed Mode

In mixed mode, SDRR is configured to both emulate and drive the systems that provide data feeds to TBFM. For the en route data, SDRR generates CMS messages for injections into an EDDS, emulating the feed that would be provided by ERAM. For terminal data, SDRR sends IFDT messages and radar data to a STARS system which then generates the AIG messages that are sent to TBFM.

In order to emulate ERAM, SDRR must be configured with an eramsim source definition and an eddserver definition with connection information for an EDDS. To connect to an EDDS server, the SDRR configuration file needs to have the "listenAddress" set to the address of the processor running SDRR. On the EDDS server, the configuration file /opt/ssr/hid_address.adp should have the same address for the E1 attribute. Both files should have the same port configured.

To drive a STARS system, the site should be added to the SDRR configuration file inside the eramsim stanza. The site should have the facility name as it is adapted in ERAM adaptation and the device should be configured for a real physical IFDT card connected to the STARS system.

```
<eramsim name="zla">
   <eddserver port="50020"  listenAddress="tbfmsdrr01"/>
   <stars name="ttt" device= "sirs16:/dev/if0" tangent="+33:47:30.41,-
   118:00:08.06" magdev="14.0"/>
</eramsim>
```

Additionally, SDRR will need configuration files for the terminal radar sites and static messages in order to generate radar data. The radar configuration files should be specified on the SDRR command line.

Terminal sensors file:

```
<root>
   <sources>
      <radar name="bur" device="/dev/srr0" type="asr9-modes" elev="813.00"
      magdev="14.00" pos="+34:12:14.80,-118:21:43.90" scantime="4.62"/>
      <radar name="lan" device="/dev/srr1" type="asr9-modes" elev="152.00"
      magdev="14.10" pos="+33:57:13.30,-118:24:28.70" scantime="4.62"/>
      <radar name="las" device="/dev/srr2" type="asr9-modes" elev="171.00"
      magdev="14.10" pos="+33:55:56.70,-118:24:25.50" scantime="4.62"/>
      <radar name="lgb" device="/dev/srr3" type="asr9-modes" elev="144.00"
      magdev="14.00" pos="+33:47:30.60,-118:00:07.00" scantime="4.62"/>
      <radar name="nfg" device="/dev/srr4" type="asr9-modes" elev="679.00"
      magdev="14.00" pos="+33:17:14.40,-117:19:50.60" scantime="4.76"/>
      <radar name="nkx" device="/dev/srr5" type="asr9-modes" elev="538.00"
      magdev="14.00" pos="+32:52:59.80,-117:08:37.90" scantime="4.62"/>
      <radar name="ont" device="/dev/srr6" type="asr9-modes" elev="961.00"
      magdev="14.00" pos="+34:03:09.00,-117:35:39.60" scantime="4.62"/>
      <radar name="psp" device="/dev/srr7" type="asr9-modes" elev="504.00"
      magdev="13.00" pos="+33:50:05.30,-116:30:22.70" scantime="4.62"/>
      <radar name="qla" device="/dev/lrr8" type="arsr3" elev="1564.00"
      pos="+33:44:45.30,-118:20:10.60" scantime="12.00"/>
      <radar name="qrw" device="/dev/lrr9" type="arsr3" elev="6236.00"
      pos="+32:52:33.60,-116:24:54.00" scantime="12.00"/>
      <radar name="qsr" device="/dev/lrr10" type="arsr3" elev="3131.00"
      pos="+35:04:56.10,-117:34:56.20" scantime="12.00"/>
      <radar name="vr1" device="multi:224.100.250.8/8250" type="asr11" elev="75.00"
      format="ast" pos="+33:58:08.50,-118:38:23.40" scantime="4.62" ttl="10"/>
      <radar name="vr2" device="multi:224.100.250.8/8250" type="asr11" elev="75.00"
      format="ast" pos="+33:58:08.60,-118:38:23.40" scantime="4.62" ttl="10"/>
      <svol name="ttt-adb" pos="+33:40:41.30,-117:52:10.14">
         <stream name="uat" device="multi:239.0.39.30/59950"/>
         <stream name="1090" device="multi:239.0.39.30/59951"/>
         <stream name="equip" device="multi:239.0.39.30/59952"/>
         <stream name="svol" device="multi:239.0.39.30/59953"/>
         <stream name="sdp" device="multi:239.0.39.30/59954"/>
      </svol>
   </sources>
</root>
```

Terminal sensor status messages file:

```
<sources>
   <radar genSectorMarks="1" name="bur" type="asr9-modes">
      <!--brtqc-->
      <static_msg>0xec0 0xdc0 0x802 0xf1f 0xff8 0x000 0xc18 </static_msg>
      <!--srtqc-->
      <static_msg>0x920 0xb40 0x80c 0xedc </static_msg>
      <!--srtqc (correlated)-->
      <static_msg>0x920 0xb40 0x80c 0xedd </static_msg>
      <!--parrot-->
      <static_msg>0x6c0 0xc93 0xa29 0x802 0x2bb 0x000 0xff7 </static_msg>
      <!--parrot-->
      <static_msg>0x6c0 0xa66 0x2df 0x802 0x2bc 0x000 0xff7 </static_msg>
      <!--permanentEcho-->
      <static_msg>0x1b0 0x080 0xaea 0xedc </static_msg>
      <!--permanentEcho (uncorrelated)-->
      <static_msg>0x1b0 0x080 0xaea 0xedd </static_msg>
      <!--permanentEcho-->
      <static_msg>0x1b0 0x100 0x2de 0xedc </static_msg>
      <!--permanentEcho (uncorrelated)-->
      <static_msg>0x1b0 0x100 0x2de 0xedd </static_msg>
      <!--status-->
      <static_msg>0x0c0 0x842 0xf03 0x800 </static_msg>
   </radar>
   ...

</sources>
```

3.1.3. Indirect Mode

For indirect mode, SDRR is configured to drive both the en route and terminal systems that provide data feeds to TBFM. For the en route data, SDRR uses SSRV command injection, interfacility messages, and radar data to drive an ERAM system which then sends CMS messages to an EDDS. For terminal data, SDRR sends radar data to a STARS system while the IFDT messages are sent to STARS by the ERAM system. STARS then generates the AIG messages that are sent to TBFM.

The en route and terminal facilities are defined in the SDRR configuration file:

```
<root>
   <comments>Generated by exportsdrrcfg 1.2.1.eng</comments>
   <sources localhost="zny">
      <arts name="aaa" device="hgi:ZNY/AAA" facName="aaa" autoTR="0">
         <hostio name="zny" facName="zcn" tangent="+40:40:25.00,-075:27:12.00"
magdev="-12.00"/>
         <artsio name="hhh" tangent="+40:11:09.00,-076:47:42.00" magdev="-11.00"/>
         <artsio name="rrr" tangent="+40:22:29.00,-075:58:13.00" magdev="-12.00"/>
         …
```

En route and terminal radar sites and static messages must be configured in order to generate radar data. Those configuration files should also be specified on the SDRR command line.

```
      <radar name="abe" device="ecgp:ZNY/ABE" type="lrr" max_psr_range="60.0"
max_ssr_range="60.0" pos="+40:40:24.77,-075:27:11.77" cv4400Converted="1"
elev="496.52" scantime="4.80"/>
      <radar name="abe" device="ecgp:ZNY/ABE" type="lrr" max_psr_range="60.0"
max_ssr_range="60.0" pos="+40:40:24.77,-075:27:11.77" cv4400Converted="1"
elev="496.52" scantime="4.80"/>
         …
      <radar name="abe" type="lrr">
      <static_msg comment="brtqc">0xec2 0x010 0x800 0x000 0xfff 0x000 0x000
</static_msg>
      <static_msg comment="srtqc">0x922 0x010 0x00f 0x020 </static_msg>
      <static_msg comment="parrot">0x6c2 0x290 0x6a7 0x802 0x2bd 0x000 0xff6
</static_msg>
   </sources>
</root>
```

A connections file is used to allow SDRR to relay the IFDT messages from ERAM to the STARS system:

```
<connections>
   <connection>
      <interfacility device="hgi:ZNY/PPP" txclock="2400" rxclock="2400" />
      <interfacility device="172.26.22.7:/dev/if0" txclock="2400" rxclock="2400" />
   </connection>
</connections>
```

3.1.4. All Modes

In addition to en route and terminal feeds, SDRR can provide other data to TBFM in all simulation modes. SDRR can be a provider of TFMS ASDI data. SDRR can be configured as an RTCS and MIS consumer with connections to a NEMS. And to facilitate development testing, SDRR can also be configured as an RTCS producer with connections to a NEMS.

Configuration for a TFMS ASDI source:

```
<tfms name="tfms" ip="" port="17010" />
```

Configuration for an MIS Consumer:

```
<tfdm name="ZLA">
   <mis device="wl://tbfmwldev01:7001?queue=jndi/TBFM_TEST_IS" />
</tfdm>
```

Configuration for an RTCS Consumer:

```
<tfdm name="KLAX">
   <rtcs publishDevice="wl://tbfmwldev01:7001?topic=RTCSPublish"
   requestDevice="wl://tbfmwldev01:7001?topic=RTCSRequest"
   unsolicitedReconWaitTime="5" hbPeriod="6" />
</tfdm>
```

Configuration for an RTCS Producer:

```
<rtcsProducer name="TBFM"
   publishDevice="wl://tbfmwldev01:7001?topic=RTCSPublish"
   requestDevice="wl://tbfmwldev01:7001?topic=RTCSRequest" />
```

4. SDRR GUI

The SDRR Graphical User Interface (GUI) is made up of a main menu bar, date and time clock, and display tabs. The menu bar includes the following options:

Windows

When multiple windows are present on a particular display tab, those windows can be arranged using options to Cascade or Tile.

Start

Starts the scenario execution, immediately.

Start At

Starts the scenario execution, at the specified time.

Help

The Help menu provides an option to select **About**. The **About** option displays the "About SDR" dialog which shows the version of SDRR, and the date and time that the SDRR executable was built.

Close

Stops the scenario execution and closes the SDRR GUI.

When SDRR is running, a runtime clock is displayed in parentheses and the current date and system time are displayed in the upper right corner. Note that this runtime clock is not synched to the start of the scenario running in the DYSIM executable, if running in dynamic mode.

The display tabs include a Status tab, and various other tabs determined by the SDRR configuration file. Right clicking in the message log areas of each display tab launches a pop-up with the following options:

Copy

Places any selected text into the copy buffer.

Select All

Selects all the text in the current display tab message log area.

Find

Opens a search bar at the bottom of the current display tab message log area.

4.1. Status Tab

The status display tab is separated into two parts. The left side shows the Device Status. All devices connected to SDRR are listed and color coded to indicate connection status. The right side contains a System Log where all system messages, such as status, warnings and errors are displayed.

4.2. Sensors Tab

The Sensors tab is displayed if radar sites or ADS-B service volumes are configured. A window for each sensor displays counts for status messages sent for each channel.

4.3. HOSTS Tab

The HOSTS tab is displayed if (non-ERAM) neighbor ARTCCs are configured. The message log area displays interfacility messages exchanged with the HOST facilities. Buttons at the top of each HOST window allow messages to be manually sent and responses to be controlled.

4.4. ARTS Tab

The ARTS tab is displayed if neighbor Terminal Sites are configured. The message log area displays interfacility messages exchanged with the Terminal facilities. Buttons at the top of each ARTS window allow messages to be manually sent and responses to be controlled.

4.5. DASI Tab

The DASI tab is displayed if configured. The message log area displays altimeter data. Buttons at the top of the window allow messages to be manually sent and the DASI value and period to be adjusted.

4.6. ETMS Tab

The ETMS tab is displayed if configured. The message log area displays TR/DT messages.

4.7. RAPPI Tab

The RAPPI tab is displayed if configured. Tabs for each configured radar are displayed within this the RAPPI tab.

4.8. ERAMSim Tab

The ERAMSim tab is displayed, if configured, and shows messages sent to and received from EDDS. A window is displayed for each ERAM facility that is included in the SDRR configuration.

Each simulated ERAM facility window has the following buttons:

Send IT

Clicking on this button causes an IT message to be sent to EDDS.

Send Msg

This button launches a dialog box where any freeform CMS message can be entered and sent to EDDS.

Show Metering Lists

This button displays the Meter Entry Viewer including meter fixes, aircraft IDs, meter times, delays and speed advisories sent by TBFM. Note that this Viewer is not updated dynamically; it must be closed and re-opened to view the most current entries.

4.9. AIG Tab

The AIG tab is displayed, if configured, and shows STARS messages exchanged with TBFM. A window is displayed for each STARS facility that is included in the SDRR configuration. These windows have a tab for each of the configured AIG datasets.

4.10. TFMS Tab

The TFMS tab is displayed if configured. This tab shows ASDI messages sent to TBFM.

4.11. TFDM Tab

The TFDM tab is displayed if any TFDM consumers are configured. If configured, an MIS window is displayed and shows metering information sent by TBFM. A window is displayed for each RTCS consumer included in the SDRR configuration and shows RTCS messages exchanged with TBFM, the RTCS Producer.

Each RTCS consumer window has the following buttons:

IssueReconRequest

Clicking on this button causes a consumer solicited reconstitution request message to be sent the RTCS Producer.

ChangeHBPeriod

This button launches a dialog box where the RTCS consumer heartbeat period can be changed. The heartbeat period is specified in seconds and controls the amount of time between heartbeat messages sent by the RTCS consumer.

ChangeUnsolicitedReconWaitTime

This button launches a dialog box where the unsolicited recon wait time can be changed. The recon wait time is specified in seconds. After receiving a heartbeat message from the RTCS Producer with a new or changed service start time, the RTCS consumer will wait the indicated amount of time for an unsolicited reconstitution message. If a reconstitution message is not received in the specified time, the RTCS consumer will send a reconstitution request to the RTCS Producer.

ChangeReceiptIgnores

This button launches a dialog box where the ReceiptIgnores value can be changed. The ReceiptIgnores value sets the number of messages received from the RTCS Producer for which the RTCS Consumer will not reply with a receipt acknowledgement message.

Show Flights

This button displays the RTCS Flight Viewer table with all of the flights received in rtcsFlt messages sent by the RTCS Producer that are applicable to the RTCS Consumer. Right clicking on an aircraft ID in the table displays options to schedule, cancel, and acknowledge a release time request. Selecting the schedule option opens a dialog where a runway and an external release time can be entered. Selecting the cancel option causes a release request message with a schedule activity of CANCEL to be sent to the RTCS Producer. Selecting the acknowledge option causes a release request message with a schedule activity of ACK to be sent to the RTCS Producer. Note that this Viewer is not updated dynamically; it must be closed and re-opened to view the most current entries.

4.12.Dynamic Input Tab

The Dynamic Input tab displays windows for Dynamic Message Status and Dynamic Target Status. The Dynamic Message Status window displays port statistics and dynamic message statistics. The Dynamic Target Status window displays port statistics, target statistics, and target status.

5. Acronyms

API	Application Program Interface
ARTCC	Air Route Traffic Control Center
ATC	Air Traffic Control
AViD	Airspace Visualization Display
CAS	Commercially Available Software
DataComm	Data Communications
DYSIM	Dynamic Simulation
EDDS	En Route Data DistributionSystem
ERAM	En Route Automation Modernization
FAA	Federal Aviation Administration
GSGT	Graphic Simulation Generation Tool
JRE	Java Runtime Environment
MIS	Metering Information Service
NAS	National Air Space
NEMS	NAS Enterprise Messaging System
RF	Radius-to-fix
RTCS	Release Time Coordination Service
RTM	Requirements Traceability Matrix
SDRR	Simulation Driver Radar Recorder
SMIF	Simulation Interface Support
STARS	Standard Terminal Automation Replacement System
TBFM	Time Based Flow Management

TFDM	Terminal Flight Data Manager
TRACON	Terminal Radar Approach Control
TSIM	TBFM Simulation
WJHTC	William J. Hughes Technical Center