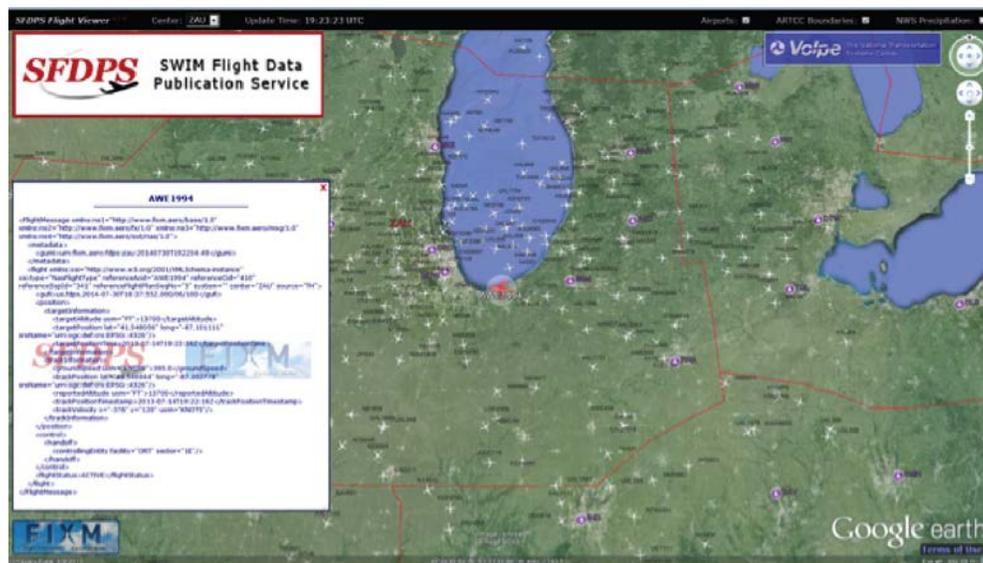


# SWIM Flight Data Publication Service (SFDPS) Connect

## User Guide

Version 2.0.2

Nov 30, 2015



Chicago Center (ZAU) Flight Data (FIXM format)

Prepared by:  
John A. Volpe National Transportation Systems Center (Volpe Center)  
Air Traffic Management Systems Division  
55 Broadway  
Cambridge, Massachusetts 02142

# Table of Contents

<b>1. Introduction</b>	<b>3</b>
<b>2. SFDPS Overview</b>	<b>3</b>
<b>3. SFDPS Data Filtering Criteria</b>	<b>5</b>
<b>4. SFDPS Data Publication Services</b>	<b>6</b>
4.1 En Route Flight Data Publication (ERFDP)	7
4.2 En Route Airspace Data Publication (ERADP)	7
4.3 En Route Operational Data Publication (ERODP)	8
4.4 En Route General Message Publication (ERGMP)	8
<b>5. SFDPS Connect Application</b>	<b>8</b>
5.1 Software Requirements	8
5.2 Execution Modes	9
5.2.1 Publish-Subscribe Service Mode	9
5.2.2 Publish-Subscribe Service Mode (no GUI)	9
5.2.3 Request-Response Mode	9
5.3 Installation	9
5.3.1 Release Package	9
5.3.2 Installation Package	10
5.3.3 Installation Instructions	11
5.4 Configuration	11
5.4.1 JMS Client Properties	12
5.4.2 Processor Properties	13
5.4.3 GUI Properties	14
5.4.4 SOAP Web Service Request Properties	15
5.4.5 Logging (log4j) Properties (Optional)	19
5.5 Execution	20
5.5.1 Using Start Script	20
5.5.2 Using Java Command	21
5.6 Running the SFDPS Connect in Execution Modes	21
5.6.1 Running Pub-Sub Mode	21
5.6.2 Running Req-Res Mode	23
5.6.3 Running Multiple SFDPS Connect Instances	24
5.7 Logging	25
5.7.1 Trace Log	26
5.7.2 Statistics Log	26
5.7.3 Data Log	26
5.7.4 Data Log (Reply Messages)	27
5.8 Possible Error Responses from Web Services Request	28
5.8.1 No Data Found	28
5.8.2 Too Much Data	28
5.8.3 Access Denied	28
<b>Appendix A: Acronyms</b>	<b>29</b>
<b>Appendix B: Sample Property Files</b>	<b>31</b>
<b>Appendix C: Manual log4j Configuration for SFDPS Connect</b>	<b>38</b>
<b>Appendix D: Differences Between v1.3 and v2.0.2</b>	<b>44</b>
<b>Appendix E: Summary of Executable Modes</b>	<b>51</b>

# 1. Introduction

This document introduces the System-Wide Information Management (SWIM) Flight Data Publication Service (SFDPS), the NextGen solution to provide users with access to real time National Airspace System (NAS) En Route information. It is written to assist users getting started with the SFDPS Connect. The User Guide consists of the following:

- [Chapter 1 Introduction](#) – current section.
- [Chapter 2 SFDPS Overview](#) – provides a brief description of SFDPS, where the data comes from and how the data is sent to users.
- [Chapter 3 SFDPS Data Filtering Criteria](#) – summarizes the criteria for the user to customize feed of SFDPS data.
- [Chapter 4 SFDPS Data Publication Services](#) – explains the data publication services supported by SFDPS.
- [Chapter 5 SFDPS Connect Application](#) - describes how to install, configure and run the SFDPS Connect, how to receive published data and how to send requests for data. It also describes how information is logged.
- [Appendix A: Acronyms](#) – describes common acronyms associated with the SFDPS project.
- [Appendix B: Sample Property Files](#) – shows sample configuration property files included in the installation package.
- [Appendix C: Manual log4j Configuration for SFDPS Connect](#) – describes in detail how to manually define log4j properties specifically for use by the SFDPS Connect application.
- [Appendix D: Differences Between v1.3 and v2.0.2](#) – describes differences between SFDPS Connect v1.3 and v2.0.2.
- [Appendix E: Summary of Executable Modes](#) – describes the various executable modes available in SFDPS Connect v2.0.2.

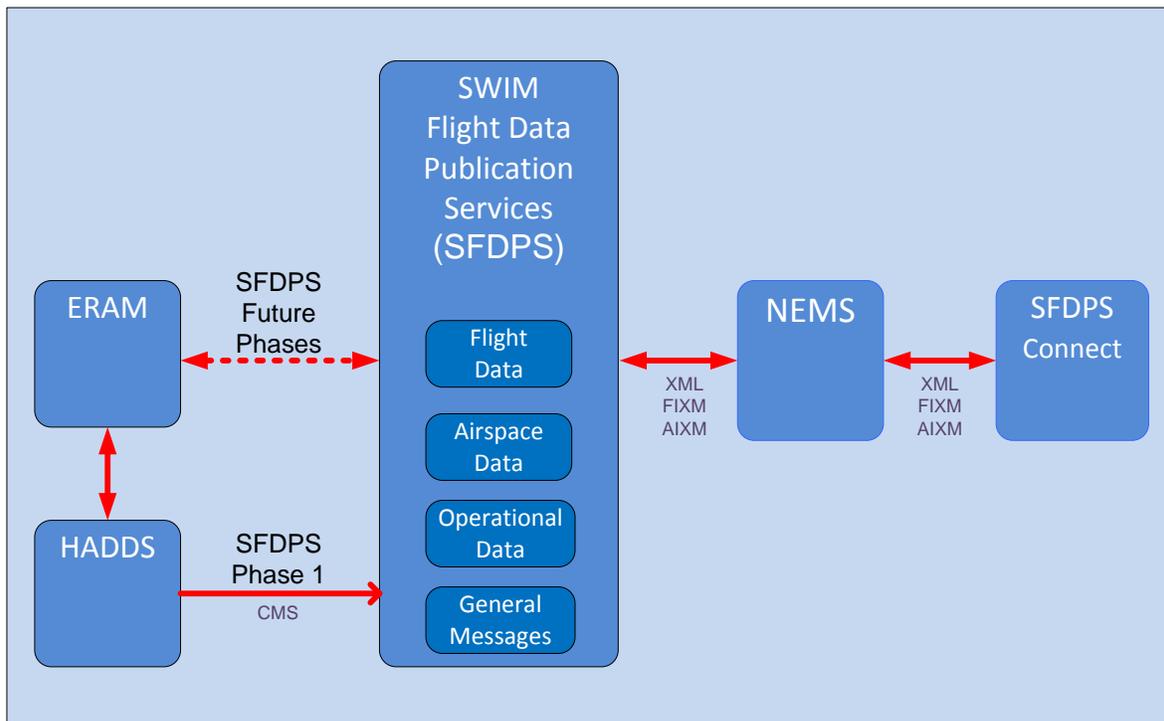
## 2. SFDPS Overview

SFDPS is a SWIM service for distributing En Route Automation Modernization (ERAM) real-time flight data through SWIM. SFDPS is an early adopter of FIXM and AIXM standards. SFDPS performs the following:

- Receives NAS en route data from ERAM Air Route Traffic Control Centers (ARTCCs) via the Host-ATM Data Distribution System (HADDS)
- Converts data from the Common Message Set (CMS) format to SWIM-compliant FIXM and AIXM, and SFDPS XML format
- Publishes data to authorized users via the NAS Enterprise Messaging Service (NEMS)
- Features:

- Flight matching
- Assigning Globally Unique Flight Identifier (GUFID)
- Customizable data flow
- Data distribution via publish-subscribe or request-response
- Data reconstitution
- Current snapshot or historical data
- Flight state (proposed, active, landed, dropped or cancelled)
- Elimination of redundant, conflicting, and non-authoritative data
- Identification and filtering of sensitive flights data

The figure below shows a conceptual view of the SFDPS system.



**Figure 2-1: Conceptual view of SFDPS**

SFDPS is hosted in two environments: Operational and Research and Development (R&D). The data in the operational environment is real time flight data. The data in the R&D environment, the Near-Live Feed (NLF), is delayed by one week.

The SFDPS Connect is a tool allowing SWIM users to access SFDPS data.

The messages published by SFDPS in FIXM format conform to the FIXM 3.0 standard. The FIXM 3.0 GUFID follows the Universally Unique Identifier (UUID) standard.

The current system architecture requires two separate installations of SFDPS, a primary and a backup; both accept data, but only one publishes it. The databases at the two sites are distinct and do not share information. As a result FIXM GUFIDs generated by each site for the same flight are different. Should a fail-over occur, the new primary site would publish a

FIXM GUFIs are different from a GUFIs generated by the old primary site for the same flight. To help deal with this situation, SFDPS creates the SFDPS-GUFIs, which are identical in both databases for the same flight. The SFDPS-GUFIs will thus allow a user to maintain a consistent sequence of messages for a flight even in a case of a failover.

### 3. SFDPS Data Filtering Criteria

SFDPS offers customizable data feeds by applying filtering criteria. This chapter contains a brief description of the four available types of filtering criteria.

#### 1. Type of Access (Message Patterns):

- **Publish-Subscribe** - The SFDPS Publish-Subscribe service provides unified, streaming flight data to authorized consumers through the SWIM NEMS infrastructure.
- **Request-Response** - The SFDPS Connect allows the user to submit data requests to SFDPS based on user defined filtering criteria. Response is generated based on user access control information.

#### 2. Data formats:

- **FIXM** – Flight Information Exchange Model, SWIM XML standard format for flight data exchange.
- **AIXM** – Aeronautical Information Exchange Model, SWIM XML standard format for airspace data exchange. Currently available only via Publish-Subscribe service.
- **Simple Schema XML** – SFDPS defined XML format (recommended for data not covered by currently used versions of FIXM and AIXM).

#### 3. Data Publication Services (more information in [Chapter 4](#)):

- Flight Data
  - Available in FIXM format (and also in Simple Schema XML)
  - Accessible via Publish-Subscribe and Request-Response
  - GUFIs and Flight Status added
  - Options:
    - **Baseline feed** – All Flight CMS messages translated to Simple Schema XML format.
    - **Enhanced feed** – Only Flight CMS messages from the authoritative source. Available in Simple Schema XML and FIXM formats

- **Airspace Data**
  - Available in AIXM (and also in Simple Schema XML)
  - Accessible via Publish-Subscribe and Request-Response (Simple Schema XML only)
- **General Messages**
  - Available in Simple Schema XML
  - Accessible via Publish-Subscribe and Request-Response
- **Operational Data**
  - Available in Simple Schema XML
  - Accessible via Publish-Subscribe and Request-Response

#### 4. Property based filtering criteria

- Data specific criteria, used for example to select particular message type, flight, airport or time frame (for Pub-Sub specified during On-ramping, for Request-Response specified as part of a request).
- Data state criteria (for Request-Response only):
  - **Current** - for Flight and Airspace (sectors or route) data
  - **Historical** - for Flight, Airspace, Operational, and General data

## 4. SFDPS Data Publication Services

SFDPS generates four different categories of data, defined as four En Route Data Services. Each of these supports the Publish-Subscribe and Request-Response.

- **En Route Flight Data Publication (ERFDP)** – Data specific to an individual flight.
- **En Route Airspace Data Publication (ERADP)** – Airspace data that is of general interest.
- **En Route Operational Data Publication (ERODP)** – Data published by ERAM to support specific FAA monitoring functions.
- **En Route General Message Publication (ERGMP)** – General, free-form text messages to one or more NAS users or classes of NAS users.

## 4.1 En Route Flight Data Publication (ERFDP)

ERFDP includes information such as proposed and active flight plans, track data associated with active flights, arrival and departure data, over-flight data and flight strip and auxiliary flight data, including runway assignments. The following list shows all ERFDP messages.

<b>Message Name</b>	<b>CMS Message Code</b>
Flight Plan Information	FH
Flight Amendment Information	AH
Converted Route Information	HX
Cancellation Information	CL
Departure Information	DH
Aircraft Identification Amendment Information	IH
Hold Information	HH
Progress Report Information	PH
Flight Arrival Information	HV
Flight Plan Update Information	HU
Expected Departure Time Information	ET
Position Update Information	HP
Tentative Flight Plan Information	NP
Tentative Aircraft Identification Amendment Information	NI
Tentative Flight Plan Removal	NL
Tentative Flight Plan Amendment Information	NU
Track Information	TH
Drop Track Information	RH
Interim Altitude Information	LH
Automated Radar Terminal System (ARTS) Flow Control Track/Full Data Block Information	HZ
Beacon Code Reassignment	BA
Beacon Code Restricted	RE
FDB Fourth Line Information	HF
Point Out Information	HT
Inbound Point Out Information	PT
Handoff Status	OH

## 4.2 En Route Airspace Data Publication (ERADP)

The ERADP service provides a combined view of airspace information that contains temporal as well as static information of sector definitions that do not change frequently. It also provides consumers with: facility sectorization assignments and updates within En Route domains; adapted arrival and departure route status; status of a route and sector (including sector configuration data); Special Activity Airspace (SAA) and altimeter settings. The following list shows all ERADP messages.

<b>Message Name</b>	<b>CMS Message Code</b>
Sector Assignment Status	SH
Route Status	HR
Special Activities Airspace (SAA)	SU
Altimeter Setting	HA

### 4.3 En Route Operational Data Publication (ERODP)

The ERODP service provides Operational data: sign-in/sign-out information, traffic count, instrument approach count adjustments, and beacon code utilization. The following list shows all ERODP messages.

<b>Message Name</b>	<b>CMS Message Code</b>
Traffic Count Adjustment	AK
Instrument Approach Count Adjustment	AC
Sign In Sign Out	SY
Beacon Code Utilization	UB
Geographic Beacon Code Utilization	UG

### 4.4 En Route General Message Publication (ERGMP)

ERGMP 'GH' messages are ad-hoc, freeform text messages entered by users of various systems. Air traffic personnel use these messages for intra-facility and inter-facility communication. Other messages in this category allow for the transmission of status information.

<b>Message Name</b>	<b>CMS Message Code</b>
General Information	GH
ERAM Status Information	HS
Unsuccessful Transmission Information	UI
Interim Altitude Status Information	HE
Hold Status Information	HO

## 5. SFDPS Connect Application

The SFDPS Connect is an application that enables a SWIM subscriber to access SFDPS data. It connects to NEMS, subscribes to receive data, and logs the received XML formatted data in files. It also sends requests to one of four SFDPS web services and receives the resulting messages through the NEMS connection. It runs on Linux.

This chapter describes how to install and run the application, how to receive the published data, and how to request data from SFDPS.

### 5.1 Software Requirements

The SFDPS Connect is a Java application which can run on the Linux platform. The following is a list of required software for running SFDPS Connect v2.0.2:

- RedHat Linux 6.1 64-bit
- Java Runtime Environment (JRE) 1.8.0\_45

- Connectivity to a NEMS server, with port open between the SFDPS Connect and the NEMS server. (Please note that connection to NEMS requires the On-Ramping procedure.)

## 5.2 Execution Modes

SFDPS Connect v2.0.2 is a Java application that can be run in three modes.

### 5.2.1 Publish-Subscribe Service Mode

The SFDPS Publish-Subscribe Service provides unified, streaming flight data to authorized consumers through the SWIM NEMS infrastructure. The GUI provides statistical information as well as shows data publication activity.

For the rest of this document, this mode will be referred to as “Pub-Sub”.

### 5.2.2 Publish-Subscribe Service Mode (no GUI)

This mode is the Pub-Sub mode without the GUI. This mode can be run in an environment without X-windows support. It still logs data and statistics similar to the regular Pub-Sub mode.

For the rest of this document, this mode will be referred to as “Pub-Sub (no GUI)”.

### 5.2.3 Request-Response Mode

The SFDPS Connect allows the user to submit data requests to SFDPS based on user defined filtering criteria. The user can request data from four data publication services provided by SFDPS: airspace data, flight data, general message, and operational data.

The user can request the following views of SFDPS data:

- Current Data – for Flight and Airspace (sectors or route) data.
- Historical Data – for Flight, Airspace, Operational, and General data.

The details of the request are pre-defined in a request property file. This mode can only be accessed through the GUI.

For the rest of this document, this mode will be referred to as “Req-Res”.

## 5.3 Installation

This section describes the release package, installation package and installation instructions.

### 5.3.1 Release Package

SFDPS Connect v2.0.2 release package contains:

1. SFDPS Connect Quick Start Guide

2. Documentation
  - SFDPS Connect User Guide (this document)
  - SFDPS Data Consumer Reference Manual
3. Installation Package (application and configuration files)
  - SFDPSConnect-v2.0.2.tar

### 5.3.2 Installation Package

The SFDPSConnect Installation Package contains the following folders:

Folder Name	Description
SFDPSConnect	Parent folder for the SFDPS Connect application.
SFDPSConnect/config	Folder containing property files used by the SFDPS Connect applications.
SFDPSConnect/data	Folder to contain logged XML data of received messages from the JMS connection.
SFDPSConnect/data-reply	Folder to contain logged XML data of received messages resulting from requests to web services.
SFDPSConnect/stats	Folder to contain statistics summary files.
SFDPSConnect/trace	Folder to contain application trace files.

The *SFDPSConnect* Installation Package contains the following files:

File Name	Location	Description
SFDPSConnect.jar	SFDPSConnect	SFDPS Connect application JAR file.
start-sfdps-connect.sh	SFDPSConnect	Script for starting the SFDPS Connect application.
sfdds-connect-pubSub.properties	SFDPSConnect/config	Property file template for SFDPS Connect running in Publish-Subscribe mode.
sfdds-connect-reqRes.properties	SFDPSConnect/config	Property file template for SFDPS Connect running in Request-Response mode.
request-airspace-data.properties	SFDPSConnect/config	Property file template for airspace data request.
request-flight-data.properties	SFDPSConnect/config	Property file template for flight data request.
request-general-message.properties	SFDPSConnect/config	Property file template for general message request.
request-operational-data.properties	SFDPSConnect/config	Property file template for operational data request.

Detailed descriptions of certain files are provided later in relevant sections of this document.

### 5.3.3 Installation Instructions

1. Copy SFDPSConnect-v2.0.2.tar into any folder of your choice. (**Note:** Ensure that this folder does not already contain a *SFDPSConnect* directory.)
2. Untar SFDPSConnect-v2.0.2.tar. This unpacks the application files into a new *SFDPSConnect* folder. The untar command is as follows:

➤ `tar xvf SFDPSConnect-v2.0.2.tar`

## 5.4 Configuration

There are five categories of configuration properties required for the SFDPS Connect. Depending on which mode the SFDPS Connect is run in, they may or may not be relevant.

Property Group	Description	Pub-Sub Mode	Pub-Sub (no GUI) Mode	Req-Res Mode
JMS Client Properties	Properties for setting up a JMS client for connecting to a topic.	X	X	X
Processor Properties	Properties related to the processing of messages where messages are analyzed and statistics are gathered and reported.	X	X	X
GUI Properties	Properties specific to run application in a GUI mode.	X		X
SOAP Web Service Request Properties	Properties required to perform SOAP web service requests. Each web service has its own property file.			X
Logging (log4j) Properties	Logging parameters.	X	X	X

Before any of the SFDPS Connect applications can be run, the property files need to be configured.

Included as part of the installation are two property files for the SFDPS Connect, one for Pub-Sub mode and one for Req-Res mode. Also included are four sample property files for web service requests. Refer to Appendix B for these samples.

This section describes in detail the configuration properties supported by the SFDPS Connect.

### 5.4.1 JMS Client Properties

This section defines the properties required to connect to a JMS topic.

Property Name	Description	Possible Values
INITIAL_CONTEXT_FACTORY	Initial context factory. This is a mandatory property.	provided by NEMS
PROVIDER_URL	URL of the data provider. This is a mandatory property.	provided by NEMS
TOPIC_CENTER_NAME	Topic to connect to. This is a mandatory property.	provided by NEMS
SECURITY_PRINCIPAL	Username. This is a mandatory property.	provided by NEMS
SECURITY_CREDENTIALS	Password. This is a mandatory property.	provided by NEMS
RECEIVE_TIMEOUT_IN_MILLIS	<p>Defines the timeout for the JMS message consumer in milliseconds. The message consumer waits to receive messages for the duration of time specified. If a message is not received during this time period, the consumer will stop waiting, assume there is a problem, and shutdown the JMS connection.</p> <p>In Pub-Sub mode, when not defined explicitly in the properties file, this value is set to a default of 120000 ms (2 minutes). After this time period, if no data is received, the application will shut down and restart the connection to the JMS topic.</p> <p>In Req-Res mode, when not defined explicitly in the properties file, this value is set to 0. This will let the consumer know not to timeout when there is no data received. In Req-Res mode, the receiving of data is dependent on when the user initiates a request, therefore it is reasonable to expect no data flow if the application is left idle. We do not want the consumer to timeout and the application to restart its connection in this situation.</p>	<p>positive integer</p> <p>Default: 120000 (in Pub-Sub mode)</p> <p>0 (in Req-Res mode)</p>
WAIT_TO_RECONNECT_TIME_IN_MILLIS	<p>Time in milliseconds to wait before a reconnection attempt to JMS is made.</p> <p>When not defined explicitly in the properties file, the application will wait a default of 30000 ms (30 seconds) before attempting a JMS reconnection attempt.</p>	<p>positive integer</p> <p>Default: 30000</p>
MESSAGE_QUEUE_SIZE	<p>Defines the size of the queue to store received messages from the topic. The JMS client component of the software will place messages received from the JMS topic onto a queue for the message processing component, which takes messages off the queue as they are processed.</p> <p>If the number of messages on the queue reaches this limit, the queue will be emptied. This is to ensure more recent received</p>	<p>positive integer</p> <p>Default: 500000</p>

	<p>messages are loaded onto the queue and the consumer does not cause the NEMS to backup.</p> <p>When not defined explicitly in the properties file, the size is set to a default of 500000.</p>	
--	--	--

## 5.4.2 Processor Properties

Processor properties to configure message processing and statistical analysis.

Property Name	Description	Possible Values
LOG_PUB_SUB_MESSAGES_IN_MAIN_LOG	<p>Sets whether or not to log accepted XML messages from Pub-Sub into the main data log file.</p> <p>This is a mandatory property.</p>	<p>true = log the messages</p> <p>false = do not log the messages</p> <p>Default: true (Pub-Sub modes)</p> <p>Default: false (Req-Res mode)</p>
LOG_REPLY_MESSAGES_IN_MAIN_LOG	<p>Reply messages are a result of a SOAP request to web services. If the SFDPS Connect is connected to a topic where these messages are being sent to, they will be received by the SFDPS Connect and this property controls whether they should be included in the main data log file.</p> <p>This is a mandatory property.</p> <p>When set to true, it tells the application to log reply messages it receives into the main data log file.</p>	<p>true = log the messages</p> <p>false = do not log the messages</p> <p>Default: false (Pub-Sub modes)</p> <p>Default: false (Req-Res mode)</p>
REPLY_LOG_FOLDER_NAME	<p>Defines the folder where reply messages are to be logged when received.</p> <p>When a message is received, it will be logged into a single file represented by timestamp and subscription ID. The files resulting from a given request as represented by the unique subscription ID will be contained in a sub-folder under this defined folder.</p>	<p>alphanumeric characters, may contain underscore and dash</p> <p>Default: data-reply</p>
STATISTICS_LOG_FOLDER_NAME	<p>Defines the folder where the statistics summary files are to be stored.</p>	<p>alphanumeric characters, may contain underscore and dash</p> <p>Default: stats</p>
STATISTICS_LOG_FILE_NAME	<p>Defines the name of the log file where statistics reports are logged. This property is mandatory.</p>	<p>alphanumeric characters, may contain</p>

Property Name	Description	Possible Values
	By default, statistics reports for the SFDPS Connect running in Pub-Sub mode are logged in files named "stats-pubSub". Statistics reports for the SFDPS Connect running in Req-Res mode are logged in files named "stats-reqRes". These files are created the "stats" folder. Periodically, the application will gather statistics and write the summary into files named <i>stats-pubSub.yyyyMMddHHmmssSSS.log</i> or <i>stats-reqRes.yyyyMMddHHmmssSSS.log</i>	underscore and dash  Default: stats-pubSub (Pub-Sub modes)  Default: stats-reqRes (Req-Res modes)
STATICTICS_LOG_START_TIME_IN_SEC	Defines when statistics report logging should start relative to when the application was run.  When not explicitly defined in the properties file, the time is set to a default of 30 seconds.	Default: 30 seconds
STATISTICS_LOG_TIME_INTERVAL_IN_SEC	Periodic time in seconds when statistics reports are to be logged.  When not explicitly defined in the properties file, the time is set to a default of 60 seconds.	positive integer  Default: 60 seconds

### 5.4.3 GUI Properties

These are properties associated with the GUI of the SFDPS Connect. All GUI related properties are optional.

Property Name	Description	Possible Values
CUSTOM_TITLE	Title to display in the GUI title bar.  When not defined explicitly in the properties file, the default title is blank.	any combination of printable characters  Default: blank
MAX_NUMBER_OF_LINES_IN_TEXT_DISPLAY	Number of lines to display in the GUI status panel before it is cleared.  When not defined explicitly in the properties file, the default of 5000 is defined.	positive integer  Default: 5000
AIRSPACE_DATA_REQUEST_PROPERTIES_FILE	Path and name of file containing airspace data request properties.	any combination of valid characters for folder and file names

Property Name	Description	Possible Values
	When not defined explicitly in the properties file, the default <b><i>config/request-airspace-data.properties</i></b> is used.	Default: config/request-airspace-data.properties
FLIGHT_DATA_REQUEST_PROPERTIES_FILE	Path and name of file containing flight data request properties.  When not defined explicitly in the properties file, the default <b><i>config/request-flight-data.properties</i></b> is used.	any combination of valid characters for folder and file names  Default: config/request-flight-data.properties
GENERAL_MESSAGE_REQUEST_PROPERTIES_FILE	Path and name of file containing general message request properties.  When not defined explicitly in the properties file, the default <b><i>config/request-general-message.properties</i></b> is used.	any combination of valid characters for folder and file names  Default: config/request-general-message.properties
OPERATIONAL_DATA_REQUEST_PROPERTIES_FILE	Path and name of file containing general message request properties.  When not defined explicitly in the properties file, the default <b><i>config/request-operational-data.properties</i></b> is used.	any combination of valid characters for folder and file names  Default: config/request-operational-data.properties

#### 5.4.4 SOAP Web Service Request Properties

To set up the filtering rules for a Request-Response service, the user must customize the properties for the selected data publication service. Each data publication service has a separate service property file containing common and service specific properties. The files provided for each type of request are:

- *request-airspace-data.properties*
- *request-flight-data.properties*
- *request-general-message.properties*
- *request-operational-data.properties*

The following table lists the supported web service request properties.

Property Name	Description	Possible Values
endpointN (where N = 0, 1, 2, ...)	The Service Endpoint tells the program where to send the request. Provided by NEMS. Used to configure NEMS connection. The number of endpoints defined depends on the number of services available for sending requests to.	http://provided_by_NEMS/FDPS/EnrouteAirspaceDataPublicationProxy
FdpsAirSpaceDataType	Type for Airspace data. Used to define Request for data.	Sector, Route or Altimeter
FdpsArrivalTimeEnd	The end of the time interval during which the flights arrived or are expected to arrive. In the format: YYYY-MM-DDThh:mm:ssZ Used to define Request for data.	2014-06-18T18:01:00Z
FdpsArrivalTimeStart	The start of the time interval during which the flights arrived or are expected to arrive. In the format: YYYY-MM-DDThh:mm:ssZ Used to define Request for data.	2014-06-18T18:01:00Z
FdpsDataFormat	The format selection for the flight data. If FIXM, then FdpsEnhancedData must be ENHANCED. Used to define Request for data.	FIXM or SIMPLEXML
FdpsDataState (this property is for Airspace data, see entry below for FdpsDataState for Flight data)	The state of the FDPS data being requested.  CURRENT requires: - Set FdpsAirspaceDataType to Sector or Route. - Set FdpsMessageType to ALL. - Set FdpSourcefacility to specific ARTCCs or ALL. - Set FdpReportingStation to nil. - Set the FdpsReceivedTimeStart and FdpsReceivedTimeEnd to nil.  HISTORICAL requires: - Set FdpsAirspaceDataType to Sector, Route or Altimeter. - Set FdpsMessageType to HA, HR, SU, SH, or ALL. These must agree with the data type: Sector (SH, SU), Route (HR), Altimeter (HA). - For Altimeter data set FdpsReportingStation to an airport name, otherwise set it to nil. - Set FdpsSourceFacility to specific ARTCCs or ALL. - Set the FdpsReceivedTimeStart and FdpsReceivedTimeEnd to bracket when SFDPS received the data.  Used to define Request for data.	CURRENT - means data on the current state of sectors or routes.  HISTORICAL - data received by SFDPS in a certain time range. SFDPS stores data for fifteen days.

Property Name	Description	Possible Values
FdpsDataState  (this property is for Flight data, see entry above for FdpsDataState for Airspace data)	<p>The state of the FDPS data being requested.</p> <p>Requires reconfiguring of the following properties: FdpsMessageType, FdpsDataFormat, FdpsEnhancedData, SourceFacility, FdpsOriginAirport, FdpsDestinationAirport, FdpsFlightIdentifier, FdpsFlightOperator.</p> <p>CURRENT request requires:  - Departure and Arrival times.  - FdpsReceivedTimeStart and FdpsReceivedTimeEnd set to nil.</p> <p>HISTORICAL request requires:  - Departure and Arrival times set to nil.  - ReceivedTimeStart and ReceivedTimeEnd set to bracket when SFDPS received the data.</p> <p>Used to define Request for data.</p>	<p>CURRENT - means the data pertains to only flights that are currently active or planned.</p> <p>HISTORICAL - means that the data, received by SFDPS within specified time, can represent all flights including active, planned and past flights that have landed or been cancelled. SFDPS stores data for fifteen days.</p>
FdpsDepartureTimeEnd	<p>The end of the time interval during which the flights departed or are expected to depart.  In the format:  YYYY-MM-DDThh:mm:ssZ</p> <p>Used to define Request for data.</p>	2014-06-18T18:01:00Z
FdpsDepartureTimeStart	<p>The start of the time interval during which the flights departed or are expected to depart.  In the format:  YYYY-MM-DDThh:mm:ssZ</p> <p>Used to define Request for data.</p>	2014-06-18T18:01:00Z
FdpsDestinationAirport	<p>Destination Airport for the flight.</p> <p>Used to define Request for data.</p>	Can be a single airport or a list of comma-separated airports or ALL
FdpsEnhancedData	<p>Indicator if data is from the authoritative source (Enhanced).  Must be set to ENHANCED if DataFormat=FIXM</p> <p>Used to define Request for data.</p>	BASIC or ENHANCED
FdpsFlightIdentifier	<p>This is the call sign, that is, the flight identifier under which the flight is operating.</p> <p>Used to define Request for data.</p>	Can be a single flight ID or a list of comma-separated flight IDs or ALL
FdpsFlightOperator	<p>The FAA-approved three-letter organizational code under which the flight is operating.  Applies only to the flight data service; applied only if the flight identifier contains a three-letter code.</p> <p>Used to define Request for data.</p>	Can be a single flight operator or a list of comma-separated operators or ALL

Property Name	Description	Possible Values
FdpsMessageType	<p>Indicates the type of CMS message.</p> <p>Used to define Request for data.</p>	<p>For Airspace data: HR, HA, SU, SH, HR_AIXM, SH_AIXM, SU_AIXM or ALL</p> <p>For Flight data: FH, AH, HX, CL, DH, IH, HH, PH, HV, HU, ET, HP, NP, NI, NL, NU, TH, RH, LH, HZ, BA, RE, HF, HT, PT, OH, FH_FIXM, AH_FIXM, HX_FIXM, CL_FIXM, DH_FIXM, IH_FIXM, HH_FIXM, PH_FIXM, HV_FIXM, HU_FIXM, ET_FIXM, HP_FIXM, NP_FIXM, NI_FIXM, NL_FIXM, NU_FIXM, TH_FIXM, RH_FIXM, LH_FIXM, HZ_FIXM, BA_FIXM, RE_FIXM, HF_FIXM, HT_FIXM, PT_FIXM, OH_FIXM,</p> <p>For General Message data: GH</p> <p>For Operational data: AC, AK, SY or ALL</p> <p>For Status message: STATUS</p>
FdpsOriginAirport	<p>Origin Airport for the flight.</p> <p>Used to define Request for data.</p>	<p>Can be a single airport or a list of comma-separated airports or ALL</p>
FdpsReceivedTimeEnd	<p>Value to end a time interval for when SFDPS received data; in the format:</p> <p>YYYY-MM-DDThh:mm:ssZ</p> <p>Used to define Request for data.</p>	<p>2014-06-18T18:01:00Z or null for a CURRENT request</p>
FdpsReceivedTimeStart	<p>Values to start a time interval for when SFDPS received data; in the format:</p> <p>YYYY-MM-DDThh:mm:ssZ</p> <p>Used to define Request for data.</p>	<p>2014-06-18T18:01:00Z or null for a CURRENT request</p>

Property Name	Description	Possible Values
FdpsReportingStation	The exact location for an Altimeter setting. It is usually an airport.  Used to define Request for data.	Airport code or nil
FdpsRequestDestinationIdentifier	The Destination Identifier is the same as the username and tells NEMS where to send replies to.	Provided by NEMS
FdpsSourcefacility	The ARTCC which triggered the message.  Used to define Request for data.	Center Code or ALL for all centers
FdpsSpecialFilters	Not used currently; for future development.	TBD
rounds	Number of times to cycle through all defined endpoints until connection to NEMS is established and a Web service request can be sent.  The software will stop connection attempts when a connection is established and request is sent or when the number of rounds specified is reached.	1 to 50.  If a value is not specified or is less than 1, the software will default to 1. If a value over 50 is entered, the software will default to 50.
timeoutinms	The timeout tells the program how long to wait for a response. Measured in milliseconds.  Used to configure SFDPS Connect.	60000
username	Username supplied by NEMS.	provided by NEMS
password	Password supplied by NEMS.	provided by NEMS

#### **5.4.5 Logging (log4j) Properties (Optional)**

SFDPS Connect v2.0.2 uses log4j to handle all logging activities. Logging is already pre-configured programmatically. Therefore, there is no need for any explicit configuration of the software in order for the logging to function.

However in case a default configuration does not support user's needs, SFDPS Connect provides ability to customize the logging set-up.

### 5.4.5.1 Default Configuration

By default, SFDPS Connect will automatically log the following:

- SFDPS messages resulting from Pub-Sub into *data-pubSub.log* files into the “data” folder. A new data file is created every minute by default.
- Trace messages are logged to the:
  - terminal console (in Pub-Sub and Req-Res modes),
  - status panel (in Req-Res mode), and to
  - trace log files (*trace-pubSub.log* in Pub-Sub mode and *trace-reqRes.log* in Req-Res mode) into the “trace” folder.

A new trace file is created every minute by default.

### 5.4.5.2 Custom Configuration

SFDPS Connect v2.0.2 allows users the ability to modify the periodic logging conditions of data and trace messages, by allowing for the explicit definition of log4j configuration properties in the *sfdps-connect-pubSub.properties* or *sfdps-connect-reqRes.properties* files.

Refer to Appendix C for details regarding how to manually define log4j configuration properties for the SFDPS Connect application.

## 5.5 Execution

There are two ways SFDPS Connect can be initiated. The predefined Start Script provides all necessary commands to start SFDPS Connect in a default set-up; however user may choose to enter his own Java commands.

### 5.5.1 Using Start Script

A start script (*start-sfdps-connect.sh*) is provided as part of the SFDPS Connect installation package. It provides a way for the user to quickly execute the application without the need to enter the Java command and accompanying command line arguments. The script uses the default property file names and paths from the installation.

To run the SFDPS Connect in Pub-Sub mode:

- `./start-sfdps-connect.sh -ps`

To run the SFDPS Connect in Pub-Sub (no GUI) mode:

- `./start-sfdps-connect.sh -psng`

To run the SFDPS Connect in Req-Res mode:

- `./start-sfdps-connect.sh -rr`

Refer to Appendix D for more information.

## 5.5.2 Using Java Command

The SFDPS Connect is a Java application: SFDPSConnect.jar.

The SFDPS Connect establishes a connection to a JMS topic to receive data through the SFDPS Pub-Sub service or receive data resulting from web services requests if the topic it is configured to connect to is defined for Request-Reply functionality.

To run the SFDPS Connect in Pub-Sub mode:

- `java -Xms512m -Xmx1024m -jar SFDPSConnect.jar -ps config/sfdps-connect-pubSub.properties`

To run the SFDPS Connect in Pub-Sub (no GUI) mode:

- `java -Xms512m -Xmx1024m -jar SFDPSConnect.jar -psng config/sfdps-connect-pubSub.properties`

To run the SFDPS Connect in Req-Res mode:

- `java -Xms512m -Xmx1024m -jar SFDPSConnect.jar -rr config/sfdps-connect-reqRes.properties`

## 5.6 Running the SFDPS Connect in Execution Modes

This section provides additional information about running the SFDPS Connect in various execution modes.

### 5.6.1 Running Pub-Sub Mode

The SFDPS Connect will attempt to connect to NEMS using the information provided by NEMS and contained in the properties file. The SFDPS Connect will establish the connection to a topic.

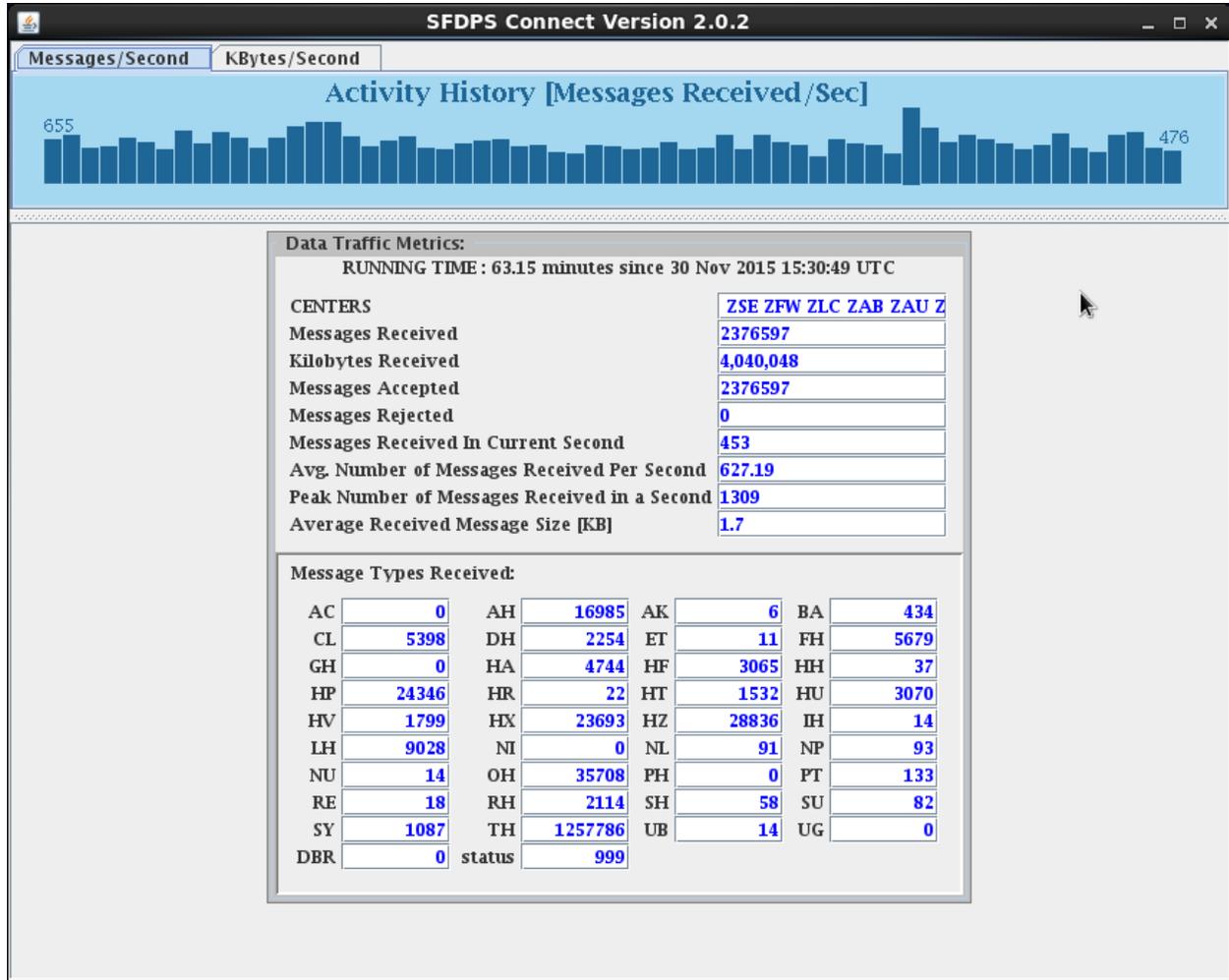
Once the published data starts streaming from the topic, the SFDPS Connect will process the data and update statistics. The SFDPS Connect will examine the general properties of each received message to determine if it is a valid SFDPS message. It checks for things such as proper data type and message type definitions. The SFDPS Connect will count all messages it receives, accepts, and rejects.

Data logs of accepted messages can grow to be very large. Each file is formatted in XML. The beginning and end of each message is marked by the SFDPSConnectMsg tag. Within this message tag are the JMS properties (JMSProps tag) and the message data proper (FDPSMsg tag).

It is important to remember that if the SFDPS Connect receives data from SFDPS in the R&D environment, the Near-Live Feed contains data that is exactly one week old. Times within messages will be a week old. Times such as when SFDPS received a message will be current.

Trace messages are logged to the terminal console in real time, and hourly into the trace log files in the trace folder. Statistics are reported hourly in the stats files located in the stats folder.

Figure 5-1 shows the GUI running in Pub-Sub mode.



**Figure 5-1: SFDPS Connect Running in Pub-Sub Mode**

Running the GUI in Pub-Sub mode displays real time statistics and activity history over a minute period for messages and Kbytes received per second. For more information on SFDPS supported message formats, refer to the *SFDPS Data Consumer Reference Manual*.

By default, messages received and accepted from Pub-Sub are logged into hourly data log files in the *data* folder. Trace messages are logged to the terminal console in real time, and

hourly into the trace log files in the *trace* folder. Statistics are reported hourly in the stats files located in the *stats* folder.

## 5.6.2 Running Req-Res Mode

When running in Req-Res mode, the user has the ability to send requests to SFDPS web services for data.

The SFDPS Connect allows the user to submit data requests to SFDPS web services based on user defined filtering criteria. Refer to Appendix B for list of all customizable properties and their descriptions. The user can request data from any of the data publication services provided by SFDPS.

The user can request the following views of SFDPS data:

- Current data – for Flight and Airspace (sectors or route) data.
- Historical data – for Flight, Airspace, Operational, and General data in a given time range that meets filtering criteria.

When a request is initiated, the SFDPS Connect will attempt to connect to NEMS a configurable number of times through specified endpoints until a connection is established. That configurable number is specified in the soap properties configuration file as the parameter *rounds*.

The SFDPS Connect sends the request to the web service including the filtering criteria for the data.

The SFDPS web service validates the incoming request and returns a unique Subscription ID to the SFDPS Connect. The Subscription ID allows for identification of which messages belong to a specific request. SFDPS sends the data that fulfills the request to a topic in a stream of messages. Each message contains the Subscription ID that identifies the request.

The SFDPS Connect may be configured to connect to a topic that receives data from a request to one of the four SFDPS web services. If reply messages are sent to the SFDPS Connect, the SFDPS Connect creates a new sub-folder in *data-reply* based on the timestamp of the request and Subscription ID. The sub-folder is created in the following format:

```
data-reply/yyyyMMddHHmmssSSS_<subscription_ID>
```

All of the received messages that match the filtering criteria are put there. Each resulting message is written to a separate file in the sub-folder. The resulting files have the file name format:

```
reply_<subscription_ID>_<sequence_number>.log
```

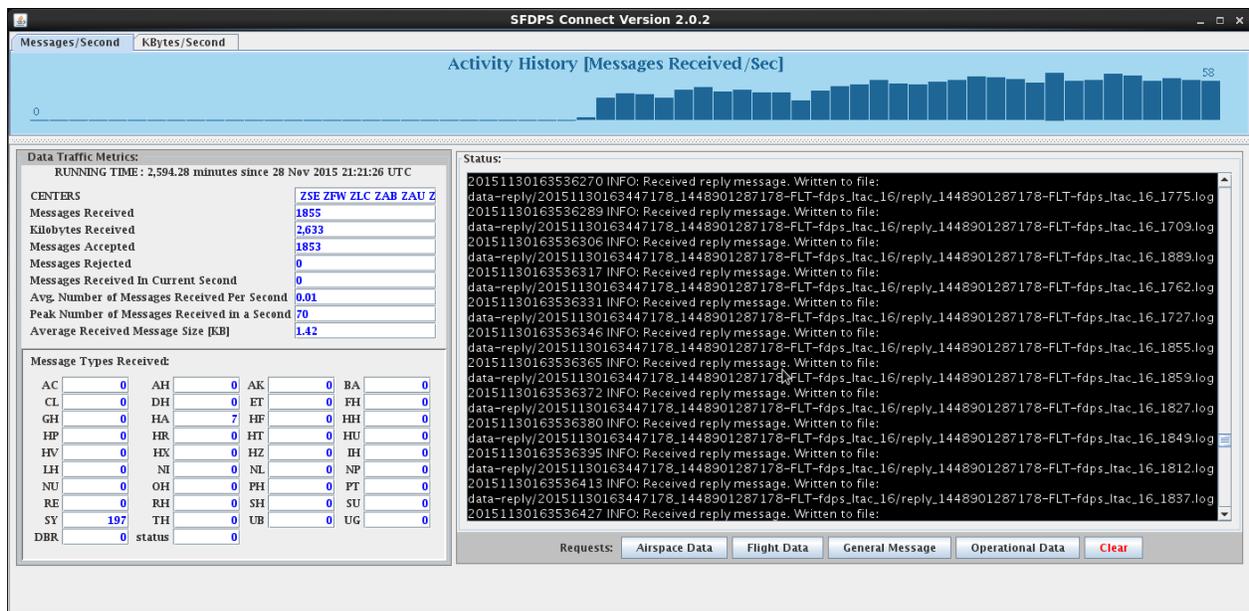
The reply messages may also be written to the main data log file but by default, this is disabled.

A request may fail or return unexpected results. Refer to Section 5.7 for details.

Trace messages are logged to the terminal console in real time, and hourly into the trace log files in the trace folder. Statistics are reported every minute (by default) in the stats files located in the stats folder.

Any trace messages resulting from the web services request are logged to the text console as well since Req-Res mode has the added feature of a status panel display in the GUI.

Figure 5-2 shows the GUI running in request-response mode.



**Figure 5-2: SFDPs Connect Running in Request-Response Mode**

A particular web services request can be made by pressing one of the four request hot buttons: **“Airspace”**, **“Flight”**, **“General Message”**, and **“Operational”**.

The gathered statistics and the black status panel can be cleared by pressing the **“Clear”** button.

### 5.6.3 Running Multiple SFDPs Connect Instances

The current installation of the SFDPs Connect is prepared for the environment to run one instance of the SFDPs Connect in Pub-Sub mode, and one instance in Req-Res mode in the same folder. All of the configuration properties and default settings are customized for this specific scenario.

If there is a need to run multiple instances of the SFDPS Connect applications simultaneously beyond the single instance of Pub-Sub mode and single instance of Req-Res mode, it is recommended that the user create separate environments.

After installation, the *SFDPSConnect* folder can be renamed and the installation can be repeated to create a new *SFDPSConnect* folder. Each individual folder would then have their corresponding property files configured accordingly to run a single instance of the SFDPS Connect in Pub-Sub mode and a single instance of the SFDPS Connect in Req-Res mode.

## 5.7 Logging

SFDPS Connect generates four types of log files depending on the mode which it is run in. The following table summarizes the logging and the default characteristics.

Log Type	Default Name and Path	Default Frequency	Execution Mode
Trace	trace/trace-pubSub.log (rolling logger active file) trace/trace-pubSub. yyyyMMddHHmm.log  or  trace/trace-reqRes.log (rolling logger active file) trace/trace-reqRes. yyyyMMddHHmm.log	Every Minute	Pub-Sub or Req-Res
Statistics	stats/stats-pubSub. yyyyMMddHHmmSSsss.log  or  stats/stats-reqRes. yyyyMMddHHmmSSsss.log	Every Minute	Pub-Sub or Req-Res
Data	data/data-pubSub.log (rolling logger active file) data/data-pubSub. yyyyMMddHHmm.log  or  data/data-reqRes.log (rolling logger active file) data/data-reqRes. yyyyMMddHHmm.log	Every Minute	Pub-Sub or Req-Res (if logging received data in the main data log is enabled)
Data (Reply Messages)	Reply messages are stored in individual files in sub-folders specific to the request based on the subscription ID. All are stored in data-reply folder.  data-reply/yyyyMMddHHmmSSsss_<sub-id>/<sub-id>.log  where: yyyyMMddHHmmSSsss = time reply message was received  e.g. data-reply/20151123232417391_1448321057391-AIR-fdps_ltac_16/reply_1448321057391-AIR-fdps_ltac_16_4.log	Periodic logging is not applicable. Only logs when data is received as a result of a request.	Req-Res

### 5.7.1 Trace Log

Logs system trace messages resulting in program execution. Typical information displayed in these log files includes:

- General program activity
- JMS connection information
- Web services request information
- Errors or exceptions

All messages logged in this file are preceded by a timestamp in the following format:

yyyyMMddHHmmSSsss

where:

yyyy = year  
MM = month ("01" to "12")  
dd = day  
HH = hour (24-hour format)  
Mm = minute  
SS = seconds  
sss = milliseconds

### 5.7.2 Statistics Log

Log statistics of data processed. The default is set to reporting statistics every minute. Reported is a snapshot of information of processed data at the time of report. Typical information displayed in these log files includes:

- Application start time
- Application running time
- Total number of messages received, accepted, rejected
- Number of messages received since last statistics report
- Average message rate per second
- Peak number of messages received in a second
- Breakdown of messages received per message type

### 5.7.3 Data Log

Logs the XML data received. Both accepted and rejected messages are logged. The actual XML message received is wrapped within "SFDPSConnectMsg" tags if the message is accepted or "SFDPSConnectMsgRejected" tags if the message is rejected. The JMS properties for the received message are wrapped within "JMSProps" tags within "SFDPSConnectMsg" or "SFDPSConnectMsgRejected". A message number (beginning at "1") from when the application is run is also logged as part of the tagged message header. This number is reset whenever the application is restarted. Finally, also logged is the time the application received the message. This is not to be confused with the times defined in the JMS properties.

All messages logged in this file are preceded by a timestamp in the following format:

yyyyMMddHHmmSSsss

where:

yyyy = year  
MM = month ("01" to "12")  
dd = day  
HH = hour (24-hour format)  
Mm = minute  
SS = seconds  
sss = milliseconds

### **5.7.4 Data Log (Reply Messages)**

Reply messages resulting from requests to web services are logged differently. Each message resulting from the request is logged into its own file and is logged exactly as it is received from the SFDPS system (not wrapped in "SFDPSConnectMsg" tags, not preceded by a logging timestamp).

By default, the reply data are logged in the *data-reply* folder, within a subfolder that specifies the specific subscription ID of the request. Within this subfolder are the individual files each containing a single reply message. The format of these log files are as follows:

data-reply/<timestamp\_of\_when\_reply\_received>\_<subscription\_id>/reply\_<subscription\_id>.log

where:

timestamp\_of\_when\_reply\_received = yyyyMMddHHmmSSsss

where:

yyyy = year  
MM = month ("01" to "12")  
dd = day  
HH = hour (24-hour format)  
Mm = minute  
SS = seconds  
sss = milliseconds

## 5.8 Possible Error Responses from Web Services Request

Some requests for data may not result in data response. A few possible outcomes are described here.

### 5.8.1 No Data Found

This condition occurs if the properties in the SFDPS Connect configuration file or in the Service Properties file are too restrictive.

One typical example is that the time range bounded by the values in **FdpsReceivedTimeStart** and **FdpsReceivedTimeEnd** may not exist in the data. This may occur if a Near Live Feed is being used.

**Solution:** Modify the properties files as necessary.

Some requests will return no data simply because there is no data to return. For example, the specified flight identifier may not exist.

**Solution:** Modify the Service Properties file as necessary.

### 5.8.2 Too Much Data

This condition occurs if the properties in the relevant Service Properties file are not sufficiently restrictive. The maximum number of messages that can be returned successfully by a "CURRENT" request is 40,000. The maximum number of messages that can be returned successfully by a "HISTORICAL" request is 3,000.

**Solution:** Modify the Service Properties file as necessary.

### 5.8.3 Access Denied

This condition can occur if the user name and/or password specified in the Service Properties file are/is not valid.

**Solution:** Modify the Service Properties file as necessary.

This condition can also occur if the same request is made (i.e., a SFDPS Connect button is clicked) within approximately one minute of an earlier button click. The software is designed to treat closely spaced requests that are the same as possible security attacks, and blocks them.

**Solution:** Wait a few minutes and try again.

## Appendix A: Acronyms

ARTCC	Air Route Traffic Control Center
ARTS	Automated Radar Terminal System
ATM	Air Traffic Management
CID	Computer ID
CMS	Common Message Set
ERADP	En Route Airspace Data Publication
ERAM	En Route Automation Modernization
ERFDP	En Route Flight Data Publication
ERGMP	En Route General Messaging Publication
ERODP	En Route Operational Data Publication
ESB	Enterprise Service Bus
FAA	Federal Aviation Administration
FIXM	Flight Information Exchange Model
GUFID	Globally Unique Flight Identifier
HADDS	Host Application Data Distribution System
HDFS	Hadoop Distributed File System
HTTP(S)	HyperText Transfer Protocol (Secure)
ICAO	International Civil Aviation Organization
IDE	Integrated Development Environment
IP	Internet Protocol
IPOP	Intermediate Point Of Presence
JMS	Java Messaging Service
NAS	National Airspace

NEMC	Network Enterprise Management Center
NEMS	NAS Enterprise Messaging Service
Pub-Sub	Publish-Subscribe
SAA	Special Activity Airspace
SFDPS	SWIM Flight Data Publication Service
SOAP	Simple Object Access Protocol
SSPID	Site Specific Plan Identifier
STARS	Standard Terminal Automation Replacement System
SWIM	System Wide Information Management
TBD	To Be Done / To be Determined
TCP/IP	Transmission Control Protocol/Internet Protocol
TFMS	Traffic Flow Management System
WSDL	Web Services Description Language
XML	eXtensible Markup Language
XSD	XML Schema Definition

## Appendix B: Sample Property Files

### sfdps-connect-pubSub.properties:

This file is used by the SFDPS Connect in both Pub-Sub and Pub-Sub (no GUI) modes.

```
#####  
#  
# SFDPS Connect Properties File  
#  
# Name: "sfdps-connect-pubSub.properties"  
#  
# Version: 2.0.2  
#  
# Description:  
# - Defines the properties for the SFDPS Connect application running in Publish-Subscribe mode.  
#  
#####  
  
#####  
# JMS Properties:  
#####  
  
# Initial Context Factory  
INITIAL_CONTEXT_FACTORY=provided_by_NEMS  
  
# Provider URL  
PROVIDER_URL=provided_by_NEMS  
  
# Topic Center Name  
TOPIC_CENTER_NAME=provided_by_NEMS  
  
# Username  
SECURITY_PRINCIPAL=provided_by_NEMS  
  
# Password  
SECURITY_CREDENTIALS=provided_by_NEMS  
  
# Time to wait during receive attempt before quitting.  
RECEIVE_TIMEOUT_IN_MILLIS=120000  
  
# Time to wait before re-attempting to connect to the JMS topic.  
WAIT_TO_RECONNECT_TIME_IN_MILLIS=30000  
  
#####  
# Processor Configuration Properties:  
#####  
  
# Log pub/sub XML messages that are received from the connected topic in the main data log files.  
LOG_PUB_SUB_MESSAGES_IN_MAIN_LOG=true  
  
# Log reply XML messages that are received from the connected topic in the main data log files.  
LOG_REPLY_MESSAGES_IN_MAIN_LOG=false  
  
# Name of the statistics summary log file.  
STATISTICS_LOG_FILE_NAME=stats-pubSub
```

## sfdds-connect-reqRes.properties:

This file is used by SFDPS Connect in Req-Res mode.

```
#####  
#  
# SFDPS Connect Properties File  
#  
# Name: "sfdds-connect-reqRes.properties"  
#  
# Version: 2.0.2  
#  
# Description:  
# - Defines the properties for the SFDPS Connect application running in Request-Response mode.  
#  
#####  
  
#####  
# JMS Properties:  
#####  
  
# Initial Context Factory  
INITIAL_CONTEXT_FACTORY=provided_by_NEMS  
  
# Provider URL  
PROVIDER_URL=provided_by_NEMS  
  
# Topic Center Name  
TOPIC_CENTER_NAME=provided_by_NEMS  
  
# Username  
SECURITY_PRINCIPAL=provided_by_NEMS  
  
# Password  
SECURITY_CREDENTIALS=provided_by_NEMS  
  
# Time to wait during receive attempt before quitting.  
# Set to 0 ms in request-response mode so that the JMS client will wait indefinitely for requested data.  
RECEIVE_TIMEOUT_IN_MILLIS=0  
  
# Time to wait before re-attempting to connect to the JMS topic.  
WAIT_TO_RECONNECT_TIME_IN_MILLIS=30000  
  
#####  
# Processor Configuration Properties:  
#####  
  
# Log pub/sub XML messages that are received from the connected topic in the main data log files.  
LOG_PUB_SUB_MESSAGES_IN_MAIN_LOG=false  
  
# Log reply XML messages that are received from the connected topic in the main data log files.  
LOG_REPLY_MESSAGES_IN_MAIN_LOG=false  
  
# Name of the statistics summary log file.  
STATISTICS_LOG_FILE_NAME=stats-reqRes
```

## Web Services SOAP Requests Property Files:

Airspace data request property file set to request CURRENT sector data for ZBW.

```
#####  
#  
# SFDPS Connect Properties File  
#  
# Name: "request-airspace-data.properties"  
#  
# Version: 1.00  
#  
# Description:  
# - Sets the properties for Airspace Data requests.  
#  
#####  
  
# Service endpoint tells the program where to send the request.  
endpoint0=http://provided_by_NEMS/FDPS/EnrouteAirspaceDataPublicationProxy  
endpoint1=http://provided_by_NEMS/FDPS/EnrouteAirspaceDataPublicationProxy  
endpoint2=http://provided_by_NEMS/FDPS/EnrouteAirspaceDataPublicationProxy  
  
# Number of times to cycle through all endpoints until a request is successfully sent.  
rounds=1  
  
# Username.  
username=provided_by_NEMS  
  
# Password.  
password=provided_by_NEMS  
  
# Number of milliseconds to wait for a response.  
timeoutinms=60000  
  
# The Destination Identifier is the same as the username and tells NEMS where to send replies to.  
FdpsRequestDestinationIdentifier=provided_by_NEMS  
  
# The DataState can be either CURRENT or HISTORICAL.  
#  
# CURRENT requests - data on the current state of sectors or routes.  
# Set AirspaceDataType to Sector or Route.  
# Set MessageType to ALL.  
# Set SourceFacility to specific ARTCCs or ALL.  
# Set ReportingStation to nil.  
# Set the ReceivedTimeStart and ReceivedTimeEnd to nil.  
#  
# HISTORICAL requests - data received by SFDPS in a certain time range.  
# SFDPS stores data for fifteen days.  
# Set AirspaceDataType to Sector, Route or Altimeter.  
# Set MessageType to HA, HR, SU, SH, or ALL.  
# These must agree with the data type: Sector (SH, SU), Route (HR), Altimeter (HA).  
# For Altimeter data set ReportingStation to an airport name, otherwise set it to nil.  
# Set SourceFacility to specific ARTCCs or ALL.  
# Set the ReceivedTimeStart and ReceivedTimeEnd to bracket when SFDPS received the data.  
#  
# e.g. This file is set to request CURRENT sector data for ZBW.  
  
FdpsDataState=CURRENT  
  
# MessageType for Airspace data can be HR, HA, SU, SH, or ALL.  
FdpsMessageType=ALL  
  
# SourceFacility is the ARTCC that sent the data or ALL for all Centers.  
FdpsSourcefacility=ZBW  
  
# The ReceivedTimeStart and ReceivedTimeEnd bracket a time interval.  
# The format must be YYYY-MM-DDThh:mm:ssZ, for example: 2014-06-18T18:01:00Z  
# Set to nil for a CURRENT request.
```

```

FdpsReceivedTimeStart=nil
FdpsReceivedTimeEnd=nil

# The ReportingStation is the exact location for an Altimeter setting. It is usually an airport.
FdpsReportingStation=nil

# The DataType can be either Sector, Route or Altimeter.
FdpsAirSpaceDataType=Sector

# SpecialFilters is not used; set it to TBD
FdpsSpecialFilters=TBD

```

Flight data request property file set to request CURRENT ZBW flights departing BOS.

```

#####
#
# SFDPS Connect Properties File
#
# Name: "request-flight-data.properties"
#
# Version: 1.00
#
# Description:
# - Sets the properties for Flight Data requests.
#
#####

# Service endpoint tells the program where to send the request.
endpoint0=http://provided_by_NEMS/FDPS/EnrouteFlightDataPublicationProxy
endpoint1=http://provided_by_NEMS/FDPS/EnrouteFlightDataPublicationProxy
endpoint2=http://provided_by_NEMS/FDPS/EnrouteFlightDataPublicationProxy

# Number of times to cycle through all endpoints until a request is successfully sent.
rounds=1

# Username.
username=provided_by_NEMS

# Password.
password=provided_by_NEMS

# Number of milliseconds to wait for a response.
timeoutinms=60000

# The Destination Identifier is the same as the username and tells NEMS where to send replies to.
FdpsRequestDestinationIdentifier=provided_by_NEMS

# The DataState can be either CURRENT or HISTORICAL.
#
# CURRENT requests - data on the current state of flights, the current flight plan and track data for all
# proposed and active flights.
#
# Set MessageType to ALL.
# Set the DataFormat to SIMPLEXML or FIXM.
# Set EnhancedData to ENHANCED.
# Set SourceFacility to specific ARTCCs or ALL.
# Set OriginAirport to specific airports or ALL.
# Set DestinationAirport to specific airport or ALL.
# Set FlightIdentifier to specific flight IDs or ALL.
# Set FlightOperator to specific three letter codes or ALL.
# Set the Departure and Arrival times to bracket the departure and arrival times of the flights.
# Set the ReceivedTimeStart and ReceivedTimeEnd to nil.
#
# HISTORICAL requests - data received by SFDPS in a certain time range. SFDPS stores data for fifteen
# days.
#

```

```

# Set MessageType to specific messages or ALL.
# Set EnhancedData to BASIC for all messages or ENHANCED for messages from authoritative centers only.
# Set the DataFormat to SIMPLEXML or FIXM. If FIXM, set EnhancedData to ENHANCED.
# Set SourceFacility to specific ARTCCs or ALL.
# Set OriginAirport to specific airports or ALL.
# Set DestinationAirport to specific airports or ALL.
# Set FlightIdentifier to specific flight IDs or ALL.
# Set FlightOperator to specific three letter codes or ALL.
# Set all the Departure and Arrival times to nil.
# Set the ReceivedTimeStart and ReceivedTimeEnd to bracket when SFDPs received the data.
#
# e.g. This file is set to perform a CURRENT request for ZBW flights departing BOS.

FdpsDataState=CURRENT

# MessageType for Flight data can be ALL.
FdpsMessageType=ALL

# The DataFormat can be SIMPLEXML or FIXM.
FdpsDataFormat=FIXM

# The EnhancedData can be BASIC or ENHANCED. Must be ENHANCED if DataFormat=FIXM.
FdpsEnhancedData=ENHANCED

# The SourceFacility is the ARTCC that sent the data or ALL for all Centers.
FdpsSourcefacility=ZBW

# The OriginAirport can be a single airport or a list of comma-separated airports or ALL.
FdpsOriginAirport=KBOS

# The DestinationAirport can be a single airport or a list of comma-separated airports or ALL.
FdpsDestinationAirport=ALL

# The FlightIdentifier can be a single flight ID or a list of comma-separated flight IDs or ALL.
FdpsFlightIdentifier=ALL

# The FlightOperator can be a single flight operator or a list of comma-separated operators or ALL.
FdpsFlightOperator=ALL

# The DepartureTimeStart and DepartureTimeEnd bracket a departure time interval.
# The ArrivalTimeStart and ArrivalTimeEnd bracket an arrival time interval.
#
# The format must be YYYY-MM-DDThh:mm:ssZ, for example: 2014-06-18T12:56:00Z

FdpsDepartureTimeStart=2014-08-21T00:00:00Z
FdpsDepartureTimeEnd=2014-08-21T12:00:00Z
FdpsArrivalTimeStart=2014-08-21T00:00:00Z
FdpsArrivalTimeEnd=2014-08-21T14:00:00Z

# The ReceivedTimeStart and ReceivedTimeEnd bracket a time interval.
# The format must be YYYY-MM-DDThh:mm:ssZ, for example: 2014-06-18T12:56:00Z
# Set to nil for a CURRENT request.

FdpsReceivedTimeStart=nil
FdpsReceivedTimeEnd=nil

# SpecialFilters is not used; set it to TBD
FdpsSpecialFilters=TBD

```

General message request properties file set to request general message data (GH message type) from ZBW for the specified time frame.

```
#####  
#  
# SFDPS Connect Properties File  
#  
# Name: "request-general-message.properties"  
#  
# Version: 1.00  
#  
# Description:  
# - Sets the properties for General Message requests.  
#  
#####  
  
# Service endpoint tells the program where to send the request.  
endpoint0=http://provided_by_NEMS/FDPS/EnrouteGeneralMessagePublicationProxy  
endpoint1=http://provided_by_NEMS/FDPS/EnrouteGeneralMessagePublicationProxy  
endpoint2=http://provided_by_NEMS/FDPS/EnrouteGeneralMessagePublicationProxy  
  
# Number of times to cycle through all endpoints until a request is successfully sent.  
rounds=1  
  
# Username.  
username=provided_by_NEMS  
  
# Password.  
password=provided_by_NEMS  
  
# Number of milliseconds to wait for a response.  
timeoutinms=60000  
  
# The Destination Identifier is the same as the username and tells NEMS where to send replies to.  
FdpsRequestDestinationIdentifier=provided_by_NEMS  
  
# General Message data is only available for HISTORICAL requests - data received by SFDPS in a certain  
time range. SFDPS stores data for fifteen days.  
#  
# Set MessageType to a specific message or list of messages.  
# Set the times to bracket when SFDPS received the messages.  
# Set SourceFacility to specific ARTCCs or ALL.  
#  
# e.g. This file is set to request General Message data (GH) from ZBW.  
  
# The SourceFacility is the ARTCC that sent the data or ALL for all Centers.  
FdpsSourcefacility=ZBW  
  
#The MessageType for General Message data can be only GH.  
FdpsMessageType=GH  
  
# The ReceivedTimeStart and ReceivedTimeEnd bracket a time interval.  
# The format must be YYYY-MM-DDThh:mm:ssZ, for example: 2014-06-18T12:56:00Z  
  
FdpsReceivedTimeStart=2014-08-28T00:00:00Z  
FdpsReceivedTimeEnd=2014-08-28T06:00:00Z  
  
# SpecialFilters is not used; set it to TBD.  
FdpsSpecialFilters=TBD
```

Operational data request properties file set to request SingIn/SingOut (SY message type) from ZBW for the specified time frame.

```
#####  
#  
# SFDPS Connect Properties File  
#  
# Name: "request-operational-data.properties"  
#  
# Version: 1.00  
#  
# Description:  
# - Sets the properties for Operational Data requests.  
#  
#####  
  
# Service endpoint tells the program where to send the request.  
endpoint0=http://provided_by_NEMS/FDPS/EnrouteOperationalDataPublicationProxy  
endpoint1=http://provided_by_NEMS/FDPS/EnrouteOperationalDataPublicationProxy  
endpoint2=http://provided_by_NEMS/FDPS/EnrouteOperationalDataPublicationProxy  
  
# Number of times to cycle through all endpoints until a request is successfully sent.  
rounds=1  
  
# Username.  
username=provided_by_NEMS  
  
# Password.  
password=provided_by_NEMS  
  
# Number of milliseconds to wait for a response.  
timeoutinms=60000  
  
# The Destination Identifier is the same as the username and tells NEMS where to send replies to.  
FdpsRequestDestinationIdentifier=provided_by_NEMS  
  
# Operational data is only available for HISTORICAL requests - data received by SFDPS in a certain time  
range. SFDPS stores data for fifteen days.  
#  
# Set MessageType to a specific message or list of messages.  
# Set the times to bracket when SFDPS received the messages.  
# Set SourceFacility to specific ARTCCs or ALL.  
#  
# e.g. This file is set to request SignIn/SignOut data (SY) from ZBW.  
  
# The SourceFacility is the ARTCC that sent the data or ALL for all Centers.  
FdpsSourcefacility=ZBW  
  
#The MessageType for Operational data can be AC, AK, SY, or ALL.  
FdpsMessageType=SY  
  
# The ReceivedTimeStart and ReceivedTimeEnd bracket a time interval.  
# The format must be YYYY-MM-DDThh:mm:ssZ, for example: 2014-06-18T12:56:00Z  
  
FdpsReceivedTimeStart=2014-08-28T00:00:00Z  
FdpsReceivedTimeEnd=2014-08-28T06:00:00Z  
  
# SpecialFilters is not used; set it to TBD.  
FdpsSpecialFilters=TBD
```

## Appendix C: Manual log4j Configuration for SFDPS Connect

SFDPS Connect 2.0.2 uses log4j v1.2.17. For more details refer to:

<https://logging.apache.org/log4j/1.2/>

To explicitly configure log4j to handle logging for SFDPS Connect, the user must define the log4j properties at the top of the SFDPS Connect property file.

For Pub-Sub Mode in the *sfdps-connect-pubSub.properties* file below, the log4j properties are bolded:

```
#####  
#  
# SFDPS Connect Properties File  
#  
# Name: "sfdps-connect-pubSub.properties"  
#  
# Version: 2.0.2  
#  
# Description:  
# - Defines the properties for the SFDPS Connect application running in Publish-Subscribe mode.  
#  
#####  
  
#####  
# LOGGING (log4j) PROPERTIES:  
#####  
  
# Define the root logger. Only log to stdout (console).  
log4j.rootLogger=INFO,stdout  
  
# "stdout" appender. Used to log messages to the console.  
log4j.appender.stdout=org.apache.log4j.ConsoleAppender  
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout  
log4j.appender.stdout.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n  
  
# "traceLogger" appender. Used to log trace messages.  
log4j.logger.com.SFDPSConnect.traceLogger=INFO,traceLogger  
log4j.appender.traceLogger=org.apache.log4j.rolling.RollingFileAppender  
log4j.appender.traceLogger.rollingPolicy=org.apache.log4j.rolling.TimeBasedRollingPolicy  
log4j.appender.traceLogger.rollingPolicy.ActiveFileName=trace/trace-pubSub.log  
log4j.appender.traceLogger.rollingPolicy.FileNamePattern=trace/trace-  
pubSub.%d{yyyyMMddHHmm}.log  
log4j.appender.traceLogger.layout=org.apache.log4j.PatternLayout  
log4j.appender.traceLogger.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n  
  
# "dataLogger" appender. Used to log received XML data messages.  
log4j.logger.com.SFDPSConnect.dataLogger=INFO,dataLogger  
log4j.appender.dataLogger=org.apache.log4j.rolling.RollingFileAppender  
log4j.appender.dataLogger.rollingPolicy=org.apache.log4j.rolling.TimeBasedRollingPolicy  
log4j.appender.dataLogger.rollingPolicy.ActiveFileName=data/data-pubSub.log  
log4j.appender.dataLogger.rollingPolicy.FileNamePattern=data/data-pubSub.%d{yyyyMMddHHmm}.log  
log4j.appender.dataLogger.layout=org.apache.log4j.PatternLayout  
log4j.appender.dataLogger.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n  
  
# Setting to false prevents dataLogger log messages from getting sent to the console.  
log4j.additivity.com.SFDPSConnect.dataLogger=false  
  
#####
```

```

# JMS Properties:
#####

# Initial Context Factory
INITIAL_CONTEXT_FACTORY=provided_by_NEMS

# Provider URL
PROVIDER_URL=provided_by_NEMS

# Topic Center Name
TOPIC_CENTER_NAME=provided_by_NEMS

# Username
SECURITY_PRINCIPAL=provided_by_NEMS

# Password
SECURITY_CREDENTIALS=provided_by_NEMS

# Time to wait during receive attempt before quitting.
RECEIVE_TIMEOUT_IN_MILLIS=120000

# Time to wait before re-attempting to connect to the JMS topic.
WAIT_TO_RECONNECT_TIME_IN_MILLIS=30000

#####
# Processor Configuration Properties:
#####

# Log pub/sub XML messages that are received from the connected topic in the main data log files.
LOG_PUB_SUB_MESSAGES_IN_MAIN_LOG=true

# Log reply XML messages that are received from the connected topic in the main data log files.
LOG_REPLY_MESSAGES_IN_MAIN_LOG=false

# Name of the statistics summary log file.
STATISTICS_LOG_FILE_NAME=stats-pubSub

```

For Request-Response Mode in the *sfdps-connect-reqRes.properties* file below, the log4j properties are bolded:

```

#####
#
# SFDPS Connect Properties File
#
# Name: "sfdps-connect-reqRes.properties"
#
# Version: 2.0.2
#
# Description:
# - Defines the properties for the SFDPS Connect application running in Request-Response mode.
#
#####

#####
# LOGGING (log4j) PROPERTIES:
#####

# Define the root logger. Only log to stdout (console).
log4j.rootLogger=INFO,stdout

# "stdout" appender. Used to log messages to the console.
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
log4j.appender.stdout.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n

```

```

# "traceLogger" appender. Used to log trace messages.
log4j.logger.com.SFDPSConnect.traceLogger=INFO,traceLogger
log4j.appender.traceLogger=org.apache.log4j.rolling.RollingFileAppender
log4j.appender.traceLogger.rollingPolicy=org.apache.log4j.rolling.TimeBasedRollingPolicy
log4j.appender.traceLogger.rollingPolicy.ActiveFileName=trace/trace-reqRes.log
log4j.appender.traceLogger.rollingPolicy.FileNamePattern=trace/trace-
reqRes.%d{yyyyMMddHHmm}.log
log4j.appender.traceLogger.layout=org.apache.log4j.PatternLayout
log4j.appender.traceLogger.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n

# "dataLogger" appender. Used to log received XML data messages.
log4j.logger.com.SFDPSConnect.dataLogger=INFO,dataLogger
log4j.appender.dataLogger=org.apache.log4j.rolling.RollingFileAppender
log4j.appender.dataLogger.rollingPolicy=org.apache.log4j.rolling.TimeBasedRollingPolicy
log4j.appender.dataLogger.rollingPolicy.ActiveFileName=data/data-reqRes.log
log4j.appender.dataLogger.rollingPolicy.FileNamePattern=data/data-reqRes.%d{yyyyMMddHHmm}.log
log4j.appender.dataLogger.layout=org.apache.log4j.PatternLayout
log4j.appender.dataLogger.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n

# Setting to false prevents dataLogger log messages from getting sent to the console.
log4j.additivity.com.SFDPSConnect.dataLogger=false

#####
# JMS Properties:
#####

# Initial Context Factory
INITIAL_CONTEXT_FACTORY=provided_by_NEMS

# Provider URL
PROVIDER_URL=provided_by_NEMS

# Topic Center Name
TOPIC_CENTER_NAME=provided_by_NEMS

# Username
SECURITY_PRINCIPAL=provided_by_NEMS

# Password
SECURITY_CREDENTIALS=provided_by_NEMS

# Time to wait during receive attempt before quitting.
# Set to 0 ms in request-response mode so that the JMS client will wait indefinitely for requested data.
RECEIVE_TIMEOUT_IN_MILLIS=0

# Time to wait before re-attempting to connect to the JMS topic.
WAIT_TO_RECONNECT_TIME_IN_MILLIS=30000

#####
# Processor Configuration Properties:
#####

# Log pub/sub XML messages that are received from the connected topic in the main data log files.
LOG_PUB_SUB_MESSAGES_IN_MAIN_LOG=false

# Log reply XML messages that are received from the connected topic in the main data log files.
LOG_REPLY_MESSAGES_IN_MAIN_LOG=false

# Name of the statistics summary log file.
STATISTICS_LOG_FILE_NAME=stats-reqRes

```

**Note:**

For SFDPs Connect to function properly, it is a requirement that the exact log4j configuration properties as shown be defined in the file, in the same order, prior to the definition of other, non-log4j properties.

The following section describe in detail, how to define log4j properties. As a way to demonstrate the manual configuration, the following logging activities will be defined:

- log trace messages to the terminal console
- log trace messages into hourly trace file
- log data messages into minute data files

**Setup the Root Logger:**

```
log4j.rootLogger=INFO,stdout
```

Set the root logger to log to stdout. The log level is set to INFO<sup>1</sup>.

**Setup Logging of Messages to the Terminal Console:**

```
log4j.appender.stdout=org.apache.log4j.ConsoleAppender
```

Defines the appender type to be a console appender. This basically configures log4j to log messages to the terminal console.

```
log4j.appender.stdout.layout=org.apache.log4j.PatternLayout
```

Defines the layout type to be a pattern layout. A pattern layout means that a pattern can be used to define the layout of the information to be logged.

```
log4j.appender.stdout.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n
```

Defines trace messages that are sent to the terminal console to be formatted with a timestamp defined by the pattern yyyyMMddHHmmssSSS.

**Setup Logging of Messages to the Trace File:**

```
log4j.logger.com.SFDPsConnect.traceLogger=INFO,traceLogger
```

Defines the trace logger. Setup to log INFO level messages.

---

<sup>1</sup> Log4j has numerous log levels. Refer to log4j documentation for details. ([www.log4j.org](http://www.log4j.org))

**log4j.appender.traceLogger=org.apache.log4j.rolling.RollingFileAppender**

Defines a rolling file appender for the trace logger. A rolling file appender logs trace messages into files that are created and closed at a defined period.

**log4j.appender.traceLogger.rollingPolicy=org.apache.log4j.rolling.TimeBasedRollingPolicy**

Defines that the rolling file appender will be time based. This means that the logging would be stored in time based periodic log files.

**log4j.appender.traceLogger.rollingPolicy.ActiveFileName=trace/trace-pubSub.log**  
**or**

**log4j.appender.traceLogger.rollingPolicy.ActiveFileName=trace/trace-reqRes.log**

Specifies the active file name of the file being logged. During the time period which this file is being actively logged to, the filename defined here will be the name of the file. Once we roll over to the next time interval, the filename will be modified to contain a timestamp based on a rolling policy defined in the next line.

**log4j.appender.traceLogger.rollingPolicy.FileNamePattern=trace/trace-pubSub.%d{yyyyMMddHHmm}.log**  
**or**

**log4j.appender.traceLogger.rollingPolicy.FileNamePattern=trace/trace-reqRes.%d{yyyyMMddHHmm}.log**

Configures the logger to log trace messages into an hourly trace file. The path and name of the trace files that are created is *trace/trace-pubSub.yyyyMMddHHmm.log* or *trace/trace-reqRes.yyyyMMddHHmm.log*. The logger will know to log to minute files because the time format specified ("yyyyMMddHHmm") has minute granularity.

**log4j.appender.traceLogger.layout=org.apache.log4j.PatternLayout**

Defines the layout of the trace logger to be a pattern layout.

**log4j.appender.traceLogger.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n**

Defines trace messages that are logged in the trace file to be formatted with a timestamp defined by the pattern *yyyyMMddHHmmssSSS*.

### **Setup Logging Data Messages to the Data File:**

**log4j.logger.com.SFDPSConnect.dataLogger=INFO,dataLogger**

Defines the data logger. Setup to log INFO level messages.

**log4j.appender.dataLogger=org.apache.log4j.rolling.RollingFileAppender**

Defines a rolling file appender for the data logger. A rolling file appender logs data messages into files that are created and closed at a defined period.

```
log4j.appender.dataLogger.rollingPolicy=org.apache.log4j.rolling.TimeBasedRollingPolicy
```

Defines that the rolling file appender will be time based. This means that the logging would be stored in time based periodic log files.

```
log4j.appender.dataLogger.rollingPolicy.ActiveFileName=data/data-pubSub.log  
or
```

```
log4j.appender.dataLogger.rollingPolicy.ActiveFileName=data/data-reqRes.log
```

Specifies the active file name of the file being logged. During the time period which this file is being actively logged to, the filename defined here will be the name of the file. Once we roll over to the next time interval, the filename will be modified to contain a timestamp based on a rolling policy defined in the next line.

```
log4j.appender.dataLogger.rollingPolicy.FileNamePattern=data/data-  
pubSub.%d{yyyyMMddHHmm}.log  
or
```

```
log4j.appender.dataLogger.rollingPolicy.FileNamePattern=data/trace-  
reqRes.%d{yyyyMMddHHmm}.log
```

Configures the logger to log data messages into minute data files. The path and name of the trace files that are created is *trace/trace-pubSub.yyyyMMddHHmm.log* or *trace/trace-reqRes.yyyyMMddHHmm.log*. The logger will know to log into minute files because the time format specified (“yyyyMMddHHmm”) has minute granularity.

```
log4j.appender.dataLogger.layout=org.apache.log4j.PatternLayout
```

Defines the layout of the data logger to be a pattern layout.

```
log4j.appender.dataLogger.layout.ConversionPattern=%d{yyyyMMddHHmmssSSS},%m%n
```

Defines data messages that are logged in the data file to be formatted with a timestamp defined by the pattern yyyyMMddHHmmssSSS. The resulting log files will contain this timestamp followed by the actual XML text of the received message, separated by a comma.

### **Suppress Logging of Data Messages to the Terminal Console:**

```
log4j.additivity.com.SFDPSConnect.dataLogger=false
```

Setting this to false prevents the messages being logged by the data logger to be sent to the terminal console. Because thousands of messages are received from SFDPS, displaying every message in the console will overwhelm the system.

## Appendix D: Differences Between v1.3 and v2.0.2

The following table lists general differences between SFDPS Connect v1.3 and v2.0.2.

Item	v1.3	v2.0.2
Names and Versions	"SFDPS Client v1.3"	"SFDPS Connect v2.0.2"
System Requirements: Compiler Version	Java 1.7.0_71	Java 1.8.0_45
Application: Runnable Files	<p>Contains one main application (<b>SFDPSClientV1.3.jar</b>) that can be configured to run in GUI or non-GUI modes.</p> <p>When running in GUI mode, can be configured to run in Pub-Sub or Request-Response modes.</p> <p>Contains four standalone, non-GUI web services Request-Response applications:</p> <p><b>sf dpsAirspaceSoapClient.jar</b>  <b>sf dpsFlightSoapClient.jar</b>  <b>sf dpsGenMsgDPSoapClient.jar</b>  <b>sf dpsOperationalDPSoapClient.jar</b></p> <p>When run separately, they perform a single web services request at the command line terminal.</p> <p>These are also launched from the GUI application using external command line scripts that run the Java command to start them.</p>	<p>Contains one main application (<b>SFDPSConnect.jar</b>) that can be configured to run in the following modes:</p> <ul style="list-style-type: none"> <li>- Pub-Sub</li> <li>- Pub-Sub (no GUI)</li> <li>- Request-Response</li> </ul>
Configuration: Property Files	<p><u>SFDPS Client (Pub-Sub Mode):</u></p> <p><i>sf dps.properties</i>  <i>log4j.properties</i></p>	<p><i>sf dps-connect-pubSub.properties</i>  <i>sf dps-connect-reqRes.properties</i>  <i>request-airspace-data.properties</i>  <i>request-flight-data.properties</i>  <i>request-general-message.properties</i></p>

Item	v1.3	v2.0.2
	<p><u>SFDPS Client (Request-Response Mode):</u></p> <p><i>sfdds.request_response.properties</i>  <i>fdps-flight-soap-client.props</i>  <i>fdps-airspace-soap-client.props</i>  <i>fdps-genmsg-soap-client.props</i>  <i>fdps-operational-soap-client.props</i>  <i>log4j.properties</i></p>	<p><i>request-operational-data.properties</i></p>
<p>Environment: Default Sub-Folders</p>	<p><u>data pub sub:</u></p> <ul style="list-style-type: none"> <li>- Stores data received from Pub-Sub.</li> </ul> <p><u>data req res:</u></p> <ul style="list-style-type: none"> <li>- Stores data received from Request-Response.</li> </ul>	<p><u>data:</u></p> <ul style="list-style-type: none"> <li>- Stores data received from Pub-Sub.</li> </ul> <p><u>data-reply:</u></p> <ul style="list-style-type: none"> <li>- Stores data received from Request-Response.</li> </ul> <p><u>stats:</u></p> <ul style="list-style-type: none"> <li>- Stores statistics summary logs.</li> </ul> <p><u>trace:</u></p> <ul style="list-style-type: none"> <li>- Stores trace logs.</li> </ul>
<p>Execution: Start Scripts</p>	<p>Start scripts used to run individual web services requests are available. These are launched from the GUI application:</p> <p><b>requestAIR.sh</b>  <b>requestFLT.sh</b>  <b>requestGEM.sh</b>  <b>requestOPR.sh</b></p> <p>Note that the GUI application itself does not have a start script.</p>	<p>Start scripts are available that can launch the three SFDPS Connect modes.</p> <p><b>start-sfdds-connect.sh</b></p>
<p>Execution: GUI vs. non-GUI modes</p>	<p>GUI mode is defined by setting a configuration property to "true". Setting to "false" disables the GUI.</p>	<p>The GUI is automatically run when the SFDPS Connect is started in Pub-Sub or Request-Response modes.</p>
<p>GUI: Removal of Unused</p>	<p>The "INPUT FLIGHT" feature was available but not actively used and</p>	<p>This functionality has been removed.</p>

Item	v1.3	v2.0.2
Functionality	was not properly functioning.	
GUI: Web Services Requests	In order to launch web service requests in GUI mode, paths to the command line scripts ( <b>requestAIR.sh, requestFLT.sh, requestGEM.sh, requestOPR.sh</b> ) to execute the standalone web service Request-Response programs must be hardcoded into the main properties file.	The requests to web services are now integrated into the SFDPS Connect application and can be initiated when the SFDPS Connect is started in Request-Response mode.
Logging: Pub-Sub and Reply Data	Data received from Pub-Sub and reply messages from the connected topic are logged in the <i>data_pub_sub</i> folder.  Reply messages from the connected topic are logged in the <i>data_req_res</i> folder.	Data received from Pub-Sub are logged in the <i>data</i> folder. Reply messages may be logged here if configured to do so.  Reply messages from the connected topic are logged in the <i>data-reply</i> folder.
Logging: Pub-Sub and Reply Data	Pub-Sub and Reply data logged into the data log files are embedded in an "SFDPSClientMsg" XML structure that contains a list of JMS properties, a message count, log time, and latency calculations.	Pub-Sub and Reply data logged into the data log files are embedded in an "SFDPSConnectMsg" XML structure that contains a list of JMS properties, a message count, and log time. There are no latency calculations represented.
Logging: Reply Data	Reply messages are logged into the <i>data_req_res</i> folder in sub-folders with the following naming format: Results- <request_type><yyyyMMdHHmmss>  They are logged in individual files with the naming format: <request_type>.HHmm-ssSSS<sequence_no>	Reply messages are logged into the <i>data-reply</i> folder in sub-folders with the naming format: yyyyMMdHHmmssSSS_<subscription_ID> >  They are logged in individual files with the naming format: reply_<subscription_ID>_<sequence_no>.log
Logging: Statistics Period	Statistics summary is only logged when data is received. If there is no data logged, there are no statistics	Statistics summary is reported independently of data logging. The period which a summary is generated is configurable. By default

Item	v1.3	v2.0.2
	<p>to report.</p> <p>Statistics summary reports information for the given timeframe associated with the particular log file. So if logging is set to a minute, the statistics summary reports the results for that minute. It does not report the overall state of the system.</p>	<p>it is set to 1 hour.</p> <p>Statistics summary reports overall state of the system and displays a snapshot of the same information presented in the GUI.</p>
Logging: Statistics Location	Statistics summary is defined at the end of the data log file embedded in an XML structure.	Statistics summary is defined in its own file and is in plain text.
Logging: Statistics Info	FIXM data is received but statistical information is not logged in statistics summary or the GUI.	FIXM data is received and statistical information are logged in the statistics summary, but not in the message statistics part of the GUI.
Logging: Statistics Info	AIXM data is received but not logged in statistics summary or the GUI.	AIXM data is received and logged in the statistics summary, but not in the message statistics part of the GUI.
Logging: Trace Info	Trace logging is a hidden feature that is disabled by default. It was intended more for debug purposes and not for user purposes.	Trace logging is by default enabled and provides general information regarding application activity such as connection attempts, communication issues, and other useful information. By default, trace logging is hourly.
Statistics: Reported Data Types	<p>Only reports statistics for the following message types in Simple Schema XML format:</p> <p>AC, AH, AK, BA, CL, DH, ET, FH, GH, HA, HF, HH, HP, HR, HT, HU, HV, HX, HZ, IH, LH, NI, NL, NP, NU, OH, PH, PT, RE, RH, SH, SU, SY, TH, UB, UG, DBR (combination of DBRTAI, DBRTRI, DBRTSI, and DBRTFPI), STATUS.</p>	<p>Reports statistics for the same Simple Schema XML message types as SFDPS Client v1.3 and also the following:</p> <p><u>Additional Simple Schema XML Message Types:</u></p> <p>HO, HE, UI, HS DBRTAI, DBRTRI, DBRTSI, DBRTFPI</p> <p>RReply</p>

Item	v1.3	v2.0.2
	<p>These message types are reported in the GUI statistics display.</p>	<p><u>AIXM Message Types:</u></p> <p>HR_AIXM, SH_AIXM, SU_AIXM</p> <p>DBRTRI_AIXM, DBRTSI_AIXM</p> <p><u>FIXM Message Types:</u></p> <p>AH_FIXM, BA_FIXM, CL_FIXM, DH_FIXM, ET_FIXM, FH_FIXM, HF_FIXM, HH_FIXM, HP_FIXM, HT_FIXM, HU_FIXM, HV_FIXM, HX_FIXM, HZ_FIXM, IH_FIXM, LH_FIXM, NI_FIXM, NL_FIXM, NP_FIXM, NU_FIXM, OH_FIXM, PH_FIXM, PT_FIXM, RE_FIXM, RH_FIXM, TH_FIXM</p> <p>DBRTFPI_FIXM</p> <p>RReply_FIXM</p> <p>These messages are NOT reported in the GUI statistics display. They are reported only in the statistics summary log. The messages reported in the GUI display are the same as those reported in the GUI display of SFDPS Client v1.3.</p>

The following table lists configuration properties from SFDPS Client v1.3 that have changed or been removed in SFDPS Connect v2.0.2. The equivalent property in SFDPS Connect v2.0.2 is shown. If there is no equivalent property, then "n/a" is indicated.

v1.3 Property	File	v2.0.2 Property	File
clearLogWindow.number.msgs	sfdps.properties sfdps.request_response.properties	MAX_NUMBER_OF_LINES_IN_TEXT_DISPLAY	by default not explicitly defined
displayGUI	sfdps.properties sfdps.request_response.properties	n/a	n/a

<b>v1.3 Property</b>	<b>File</b>	<b>v2.0.2 Property</b>	<b>File</b>
initial.context.factory	sfdds.properties sfdds.request_response. properties	INITIAL_CONTEXT_FACTORY	sfdds-connect- pubSub.properties  sfdds-connect- reqRes.properties
input.endpoint	sfdds.properties sfdds.request_response. properties	n/a	n/a
logMsgs	sfdds.properties sfdds.request_response. properties	LOG_PUB_SUB_MESSAGES_IN_M AIN_LOG	sfdds-connect- pubSub.properties sfdds-connect- reqRes.properties
output.file.time.length*	sfdds.properties sfdds.request_response. properties	log4j.appender.dataLogger .rollingPolicy.FileNamePa ttern	by default not explicitly defined
output.fileName.dir*	sfdds.properties sfdds.request_response. properties	log4j.appender.dataLogger .rollingPolicy.FileNamePa ttern	by default not explicitly defined
output.fileName.root*	sfdds.properties sfdds.request_response. properties	log4j.appender.dataLogger .rollingPolicy.FileNamePa ttern	sfdds-client- pubSub.properties sfdds-client- reqRes.properties
output.mmap.file	sfdds.properties sfdds.request_response. properties	n/a	n/a
output.mmap.size	sfdds.properties sfdds.request_response. properties	n/a	n/a
provider.url	sfdds.properties sfdds.request_response. properties	PROVIDER_URL	sfdds-connect- pubSub.properties sfdds-connect- reqRes.properties
request.AIR	sfdds.properties sfdds.request_response. properties	n/a	n/a
request.FLT	sfdds.properties sfdds.request_response. properties	n/a	n/a
request.GEM	sfdds.properties sfdds.request_response. properties	n/a	n/a
request.OPR	sfdds.properties sfdds.request_response. properties	n/a	n/a
request.ReplyDirRoot	sfdds.properties sfdds.request_response. properties	REPLY_LOG_FOLDER_NAME	by default not explicitly defined
request.services	sfdds.properties sfdds.request_response. properties	n/a	n/a
security.credentials	sfdds.properties sfdds.request_response. properties	SECURITY_CREDENTIALS	sfdds-connect- pubSub.properties sfdds-connect-

<b>v1.3 Property</b>	<b>File</b>	<b>v2.0.2 Property</b>	<b>File</b>
			reqRes.properties
security.principal	sfdds.properties sfdds.request_response. properties	SECURITY_PRINCIPAL	sfdds-connect- pubSub.properties sfdds-connect- reqRes.properties
topic.center.name	sfdds.properties sfdds.request_response. properties	TOPIC_CENTER_NAME	sfdds-connect- pubSub.properties sfdds-connect- reqRes.properties
window.title	sfdds.properties sfdds.request_response. properties	CUSTOM_TITLE	by default not explicitly defined

\*The functionality of `output.fileName.dir`, `output.fileName.root`, and `output.file.time.length` are covered by the single log4j property: `log4j.appender.dataLogger.rollingPolicy.FileNamePattern`. This log4j property defines the folder, filename, and period of the logging.

The details of configuration are covered in Chapter 5: SFDPS Connect Application.

## Appendix E: Summary of Executable Modes

<b>Execution Mode:</b>	Pub-Sub
<b>Description:</b>	SFDPS Connect application to perform Pub-Sub functionality, display real time statistics and activity, and log information.
<b>Executable JAR:</b>	SFDPSConnect.jar
<b>Executable Component:</b>	SFDPSConnect
<b>Properties Files:</b>	sf dps-connect-pubSub.properties
<b>Run Command (Start Script):</b>	start-sfdps-connect.sh -ps
<b>Run Command (Java):</b>	java -Xms512m -Xmx1024m -jar SFDPSConnect.jar -ps config/sfdps-connect-pubSub.properties

<b>Execution Mode:</b>	Pub-Sub (no GUI)
<b>Description:</b>	SFDPS Connect application to perform Pub-Sub functionality and log information.
<b>Executable JAR:</b>	SFDPSConnect.jar
<b>Executable Component:</b>	SFDPSConnect
<b>Properties Files:</b>	sf dps-connect-pubSub.properties
<b>Run Command (Start Script):</b>	start-sfdps-connect.sh -psng
<b>Run Command (Java):</b>	java -Xms512m -Xmx1024m -jar SFDPSConnect.jar -psng config/sfdps-connect-pubSub.properties

<b>Execution Mode:</b>	Request-Response
<b>Description:</b>	SFDPS Connect application to perform SOAP web services requests, display real time statistics and activity, and log information.
<b>Executable JAR:</b>	SFDPSConnect.jar
<b>Executable Component:</b>	SFDPSConnect
<b>Properties Files:</b>	sf dps-connect-reqRes.properties request-airspace-data.properties request-flight-data.properties request-general-message.properties request-operational-data.properties
<b>Run Command (Start Script):</b>	start-sfdps-connect.sh -rr
<b>Run Command (Java):</b>	java -Xms512m -Xmx1024m -jar SFDPSConnect.jar -rr config/sfdps-connect-reqRes.properties