Aviation Forecast Preparation System

System and User Guide

AvnFPS Team, Product Generation Branch, Meteorological Development Laboratory

Aviation Forecast Preparation System: System and User Guide

by AvnFPS Team, Product Generation Branch

Published 16 November 2009

Table of Contents

Introduction ix
1. System Administration 1
    1. System Overview 2
        1.1. Name and Event Server 4
        1.2. Data Ingest Server 5
        1.3. Data Request Server 6
        1.4. Transmission Server 7
        1.5. Client Programs 9
    2. Configuring AvnFPS 10
        2.1. Files in `etc` 11
        2.2. TAF configuration files 18
        2.3. X resources configuration files 25
    3. Starting and Stopping AvnFPS 27
        3.1. Starting AvnFPS Servers 28
        3.2. Stopping AvnFPS Servers 30
    4. Avnsetup 31
        4.1. Editing Monitoring Rules 32
        4.2. Editing TAF Site Information 35
        4.3. Editing TAF Site Templates 38
        4.4. Editing TAF Product Definition 39
        4.5. Creating Triggers 41
        4.6. Climate Data File Tool 43
        4.7. Using Text Editor 52
    5. Configuring IFPS to send gridded data to AvnFPS 53
    6. Logging 54
    7. Troubleshooting 56
        7.1. Overall Server Health 57
        7.2. Log Files 59
        7.3. Data Ingest 60
        7.4. Product Transmission 61
2. User's Guide 63
    1. Startup Menu 64
    2. TAF Monitoring 66
        2.1. TAF Monitoring GUI, aka "AvnWatch" 67
        2.2. Server Status Indicators 70
        2.3. Transmission Queue 71
        2.4. Menu Bar 73
        2.5. Resource Editor 75
        2.6. Editor and Product Selection Buttons 78
    3. TAF Editor 79
        3.1. Starting the TAF Editor 80
        3.2. Using the TAF Viewer 81
        3.3. Using the TAF Editor 82
        3.4. Using the Guidance Viewer 87
        3.5. TAF Editor Menu Bar 90
    4. Forecast Transmission 93
    5. Weather Plot 94
    6. Climatology Tools 96
        6.1. Climatology Startup Menu 97
        6.2. METAR Display 98
        6.3. Wind Rose 99
        6.4. Ceiling and Visibility Distribution 104
        6.5. Ceiling and Visibility Trend 108

A. Editor Key Bindings 111
B. Monitoring Rules 113
C. Climate Quality Control 119
D. Low Level Wind Shear Monitoring 123
E. Ceiling and Visibility Grid Monitoring 125
F. Writing AvnFPS TAF Edit Tools 127
Bibliography 131

List of Tables

A.1. Key Bindings 111
B.1. METAR Monitoring Rules 113
B.2. GFE Grid Monitoring Rules 116
B.3. Lightning monitoring rules 117
B.4. Lightning probability forecast monitoring rules 117
B.5. CCFP monitoring rules 118
B.6. Low Level Wind Shear Monitoring Rules 118
F.1. Argument List Description 127
F.2. Description of the Tools in `/toolpy` directory 129

Introduction

The Aviation Forecast Preparation System (AvnFPS) is an AWIPS application that helps forecasters prepare their aviation forecasts and monitor them in real-time using a variety of observational sources. The AvnFPS monitoring capability gives forecasters quick and continuous feedback on TAFs as well as associated observations. This monitoring capability uses a color-coded, user-configurable scheme. AvnFPS includes a specialized editor to aid the production of the TAFs. The editor can display TAFs, guidance products, and current observations as both text and graphics. AvnFPS also includes tools that help forecasters evaluate the quality of each forecast before it is issued. These quality control tools can assess the correctness of the forecast's syntax as well as compare its meteorological content with a climatological database.

Chapter 1. System Administration

This chapter is intended for those who will install, configure and maintain AvnFPS.

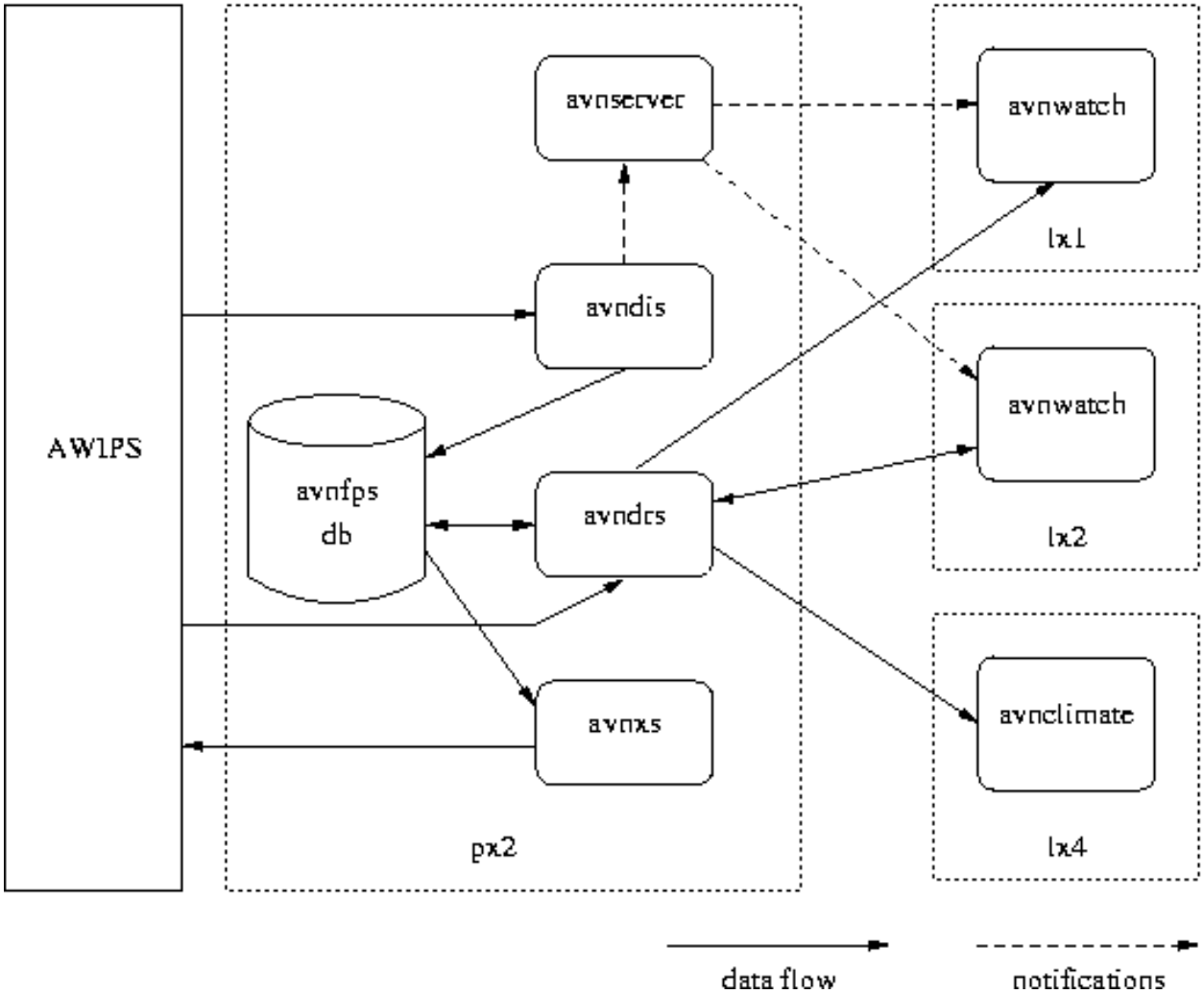
1. System Overview

This section describes technical details of AvnFPS.

AvnFPS is implemented as a collection of server processes with distributed client processes that communicate via TCP sockets. The servers can run on one or more hosts where the data are readily available. Clients run on separate, individual workstations and do not need direct access to the data. The data request server (**avndrs**) is responsible for delivering data to clients. The data ingest server (**avndis**) monitors data received by AWIPS and stores them in a modified format in the AvnFPS database. A notification system (**avnserver**) is responsible for alerting the clients when new data are received. Even the AvnFPS database can be distributed among several hosts.

AvnFPS is written in the Python programming language [Python]. Remote communication features are implemented with the help of Python Remote Objects (PYRO). PYRO [Pyro] is an advanced Distributed Object Technology system, written entirely in Python, which resembles Java™ [Java] Remote Method Invocation (RMI). The TAF and METAR decoders utilize the flexible Toy Parser Generator, [TPG], also a Python module. Graphical interfaces are coded using the Tkinter package. The AvnFPS distribution comes with its own version of the above tools, and is independent of AWIPS COTS, although much, if not all, of this will change with the delivery of AWIPS II.

The standard AvnFPS configuration is shown below.



Data flow diagram

Note

The above diagram shows servers running on `px2`, however, any Red Hat™ host on a AWIPS network would do. The standard installation puts the servers on `px2`.

1.1. Name and Event Server

The **avnserver** process stores the names and locations of all objects used by AvnFPS servers, and it provides these names and locations to all other AvnFPS processes. Additionally, **avnserver** other task is to provide a notification mechanism between servers and clients. Thus, **avnserver** combines PYRO's name and event servers. One and only one instance of avnserver should run within a WFO cluster. A client first connects to **avnserver** to get information about the locations of the data request server and to subscribe for notifications. An access list of valid hosts, maintained in a configuration file, is used to provide access control. **avnserver** must be running before any other AvnFPS servers or clients can be successfully started.

The application **avninit** is used to launch AvnFPS servers and in a certain order. Users typically do not have to be concerned about the servers. It is **avninit's** job to ensure that all servers are running and restart them if they fail.

1.2. Data Ingest Server

An instance of **avndis** monitors AWIPS data directories using the **gamin** utility [Gamin]. The **avndis** server is actually a collection of threads: for each data source, there is a dedicated thread that processes the data. Which server instance processes which data is determined by the server configuration file `etc/server.cfg` (page11). **avndis** receives a notification from the **gamin** process when a file in a monitored directory is created or modified. The **avndis** server, or parent process, then passes the notification to appropriate thread. The thread then processes the file's content, writes data to AvnFPS database and returns information back to the parent process. The parent process then sends notification to **avnserver**. A client that subscribed to the **avnserver** will then receive this notification. Currently implemented threads are:

text	Processes text products: TAFs, METARs and CCFPs.
ltg	Processes lightning data.
rltg	Processes regression-based thunderstorm forecast data.
llws	Calculates Low Level Wind Shear values. The data sources are aircraft, radar and profiler VWP data and METAR observations.
file	Monitors files written by an external program in a format that can be processed directly by AvnFPS clients. Currently used to process IFPS grids.
guid	Monitors directories where guidance sources arrive over the AWIPS WAN. Typically, BUFR MOS and raw model output from NCEP's NAM-WRF model.

Every 30 seconds **avndis** sends an ALIVE notification that is used by clients to monitor the state of AvnFPS servers and to alert users if one or more quits functioning.

1.3. Data Request Server

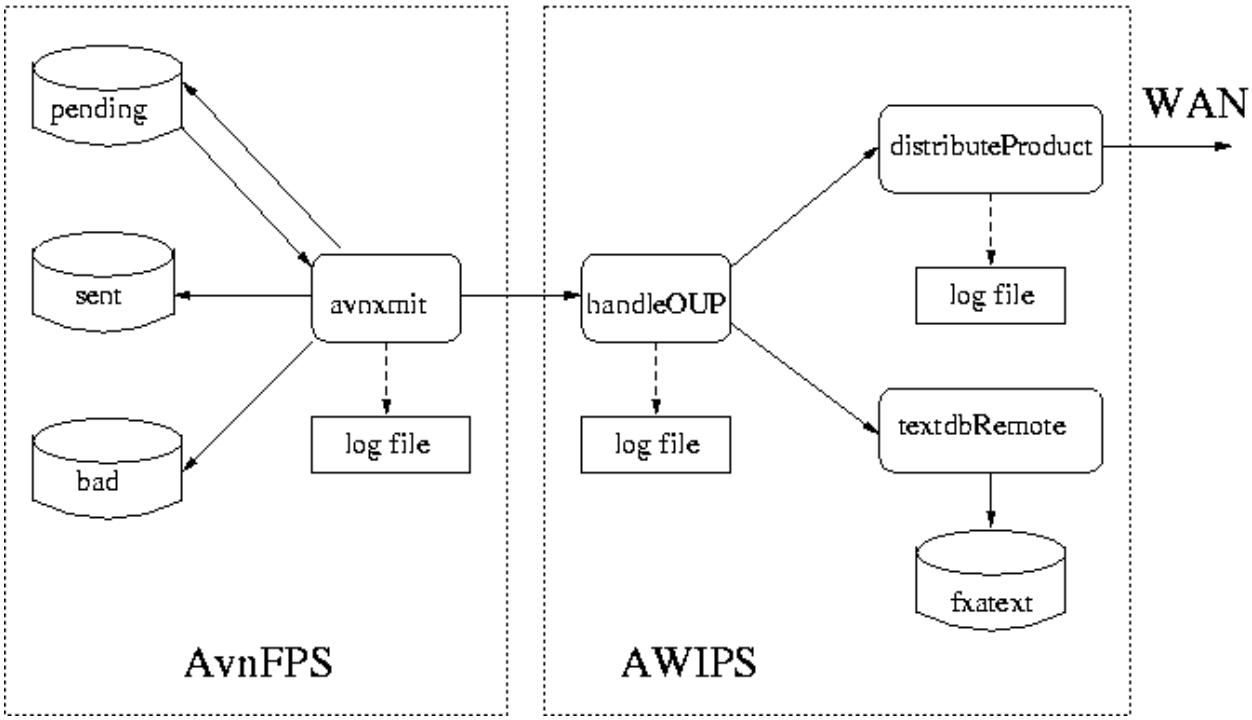
avndrs server is mainly responsible for providing data to AvnFPS clients. The data source is the AvnFPS database, however, it can also directly access AWIPS files such as NetCDF data in the `/data/fxa` tree as needed. The other function of **avndrs** is to write products prepared by the clients to a queue where they are then processed by the transmission server **avnxs**. Each instance of **avndrs** is capable of processing all data sources.

Every 30 seconds **avndrs** sends an ALIVE message that is used by clients to monitor state of AvnFPS servers and alert users if one or more quits functioning.

1.4. Transmission Server

The transmission server, **avnxs**, provides a "bridge" between AvnFPS and the AWIPS transmission system. **avnxs** supports delayed transmissions, i. e., issuing forecasts that have been prepared by the forecaster before the transmission window opens.

The data flow diagram below illustrates the read and write activities of **avnxs** and **handleOUP.pl**.



Data flow diagram: Transmission Server

Forecasts prepared by the AvnFPS forecast editor are written to the directory `xmit/pending` as text files with a particular naming convention. The filename is then used to determine the disposition of the product. Files in the `xmit/pending` directory are named as follows:

`fid-ccccnnnxxx-wmoid-wfoid-yymmddhhmm-typ-tttttttttt`
where

fid:
: Forecaster ID Number

ccccnnnxxx:
: AWIPS id, argument to **handleOUP.pl**

wmoid:
: WMO Header

wfoid:
: WMO ID of the WFO issuing the product

yymmddhhmm:
: Full timestamp, as year (modulo 100), month (Jan=1), day of the month, UTC hour rounded back to beginning of the issued hour in accordance with WMO directives for aerodrome forecasts.

typ:

Forecast Type and Version, one of the following

___	Routine Forecast
RR*X*	Delayed Forecast
AA*X*	Amendment
CC*X*	Correction

where *X* is the version: A through Z.

tttttttttt:

Earliest transmission time, in Unix seconds. For routine forecasts, this is the start of the transmission window. For other forecasts it is the time when the forecaster presses the **Send** button in the TAF Editor, or a time specified in the Send Dialog.

avnxs server monitors the files in the `xmit/pending` directory to the current system clock. The requested transmission time, T_{xmit}, and the creation time, T_{post}, are retrieved from the filename and from the OS, respectively. If the current time T_{cur} is within range: $T_{xmit} <= T_{cur} < T_{post} + N$ hours, **avnxs** will call an AWIPS program (currently **handleOUP.pl**) to send the forecast out to the Wide Area Network (WAN). *N*, the number of hours before a pending file is considered 'old', is now a configurable item in `etc/xmit.cfg` (page17) file. If the forecast file is too old, it will be moved to the `xmit/bad` directory. The value of *N* (usually 3 hours) is set in transmission server configuration file. Files that do not match transmission file format are discarded silently. **avnxs** checks for the return code from **handleOUP.pl**. The status is written to a file `xmit/`*DoW*, where *DoW* is a 3-letter abbreviation for the Day Of the Week. Here is a sample entry from `xmit/`*DoW*:

SUCCESS 015-KPBZTAFPIT-FTUS41-KPBZ-0810061100-___-1223292300
The first word is either SUCCESS or FAIL*nnn*, and *nnn* is the error code returned by **handleOUP.pl**. If **handleOUP.pl** fails to transmit the forecast, the file will be moved to `xmit/bad` directory. The transmission status will be announced via **avnserver** to all client GUIs, see Section 4, “Forecast Transmission” [93].

The transmission server also collects information on forecasts sent by the office for verification purposes. A special product is sent periodically to the Performance Branch within the Office of Climate, Water and Wather Services at NWS Headquarters. This product contains information that identifies which forecaster is responsible for which forecast. A configuration file `etc/xmit.cfg` (page17) specifies WMO and AWIPS headers and the frequency of transmission of this verification product.

1.5. Client Programs

AvnFPS Monitoring GUI, as known as **AvnWatch** is the main client application. It provides forecast preparation and monitoring functionality. **Avnclimate** is a set of GUIs accessing archived climatological data. Current functionality provides display of historical data and statistics that can help by producing a short-term forecast based on current weather conditions. These applications are described in much greater detail in Chapter 2.

2. Configuring AvnFPS

This section describes directory layout, configuration files used by AvnFPS and steps needed to download climatological data. Generally, this section will only be used as an aid in troubleshooting problems with the configuration of AvnFPS. While it is possible to configure AvnFPS by editing these files directly, most will find it much easier to use the Graphical User Interfaces (GUI) described in Section 4, “Avnsetup” [31].

After successful installation of AvnFPS OB9.2, the top level directory `/awips/adapt/avnfps` should be as follows:

```
lx2-wfo: /bin/ls -la
drwxrwxr-x   9 fxa     fxalpha 4096 Feb 21 15:04 OB9
drwxrwxr-x   9 fxa     fxalpha 4096 Mar  1 13:03 OB9.2
lrwxrwxrwx   1 root    root       7 Mar  1 13:33 bin -> OB9.2/bin
lrwxrwxrwx   1 root    root       7 Mar  1 13:33 etc -> OB9.2/etc
drwxrwxr-x   5 fxa     fxalpha 4096 Mar 24 13:03 gamin-0.1.10
drwxrwxr-x   6 fxa     fxalpha 4096 Feb 21 20:27 hdf5-1.6.4
drwxrwxr-x   7 fxa     fxalpha 4096 Mar 13  2005 Python-2.4.1
drwxrwxr-x   6 fxa     fxalpha 4096 Mar 13  2005 tcltk8.4.5
```

The installation process does attempt to remove versions of the software older than OB9. It is up to you to remove any remaining cruft that might be in this directory.

All configuration files are located in the directory `/awips/adapt/avnfps/OB9.2/etc` and its subdirectories. Most of the configuration files are in Windows™ ini format:

```
[section]

          keyword=argument(s)
```

Lines starting with # are comments.

Files containing specifications for graphical user interfaces (GUI) follow a X11 standard:

```
*pattern: value
```

where *`pattern`* is a hierarchy of one or more widget classes/names, separated by `*` or a `.`. Comment lines start with `!`. See [Welch] [131], Chapter 25. These files are located in the subdirectory `app-resources`.

2.1. Files in `etc`

`etc/forecasters`

Contains a list of forecaster numbers with their names or initials. The numbers must be in the range 1 through 999. Lines starting with # are comments. Blank lines are allowed in the file. The transmit flag which follows the forecasters identification number is a simple binary flag: 0 - no; 1 - yes. An example:

```
# etc/forecasters
# List of forecasters
# Format:
# forecaster's number xmit_privilege [No=0,Yes=1] forecaster's name
1 1 Amanda
2 0 Dan
3 1 Mark
4 1 Matt
```

So in this example Amanda, Mark and Matt have the ability to queue up TAFs for transmission, however, Dan does not.

`etc/logging.cfg`

This file configures the logging behavior of all AvnFPS programs. You should not need to modify it.

```
# logging.cfg
# configuration file for Python logging utility

[loggers]
keys=root

[handlers]
keys=hdl

[formatters]
keys=fmt

[logger_root]

level=INFO
handlers=hdl

[handler_hdl]

class=WeeklyFileHandler
args=('%s/logs' % os.environ['TOP_DIR'], os.path.basename(sys.argv[0]).split('.')[0])
formatter=fmt

[formatter_fmt]

format=%(asctime)s %(levelname)-5s %(process)5d %(module)s: %(message)s
```

1. Log level is set in the section [logger_root]. Available values in decreasing order of information output are: DEBUG or NOTSET (these two are equivalent), INFO, WARNING, ERROR, CRITICAL.
2. [handler_hdl] specifies output directory and file name. The class `WeeklyFileHandler` appends an abbreviated day of the week to file name.
3. [formatter_fmt] specifies format of a line written to the log file. Refer to Python Documentation [http://docs.python.org/lib/module-logging.html] for details.

`etc/server.cfg`

Specifies values used and directories monitored by the data ingest threads.

```
# etc/server.cfg
# Name servers
# Either local or ip address. Local means that broadcast is used to locate
# the server, which means the server must be on the same subnet
# The list of tafs is displayed in the startup (menu) GUI
```

```
# Data ingest servers
[dis]
tags=text,ltg,rltg,file,llws,guid

#[dis_text]

#name=text
#source=/data/fxa/point/avnfps/raw
#module=TextThread
#nhours=24

[dis_text]

name=text
source=/awips/adapt/avnfps/data/text
module=TriggerThread
nhours=24

[dis_ltg]

name=ltg
source=/data/fxa/point/binLightning/netcdf
module=LtgThread
# distance in miles, age in minutes
distance=20
age=15

[dis_rltg]

name=rltg
source=/data/fxa/point/gfslamp/netcdf
module=LAMPLtgThread

[dis_llws]

name=llws
radar=/data/fxa/radar
profiler=/data/fxa/point/profiler/netcdf
acars=/data/fxa/point/acarsProfiles/netcdf
module=LLWSThread

[dis_file]

name=file
source=/awips/adapt/avnfps/data/grids,
module=FileThread

[dis_guid]

name=guid
nam=/data/fxa/point/model/ETA/netcdf
nammos=/data/fxa/point/mos/ETA/netcdf
gfsmos=/data/fxa/point/mos/GFS/netcdf
gfslamp=/data/fxa/point/gfslamp/netcdf
module=GuidanceThread
nhours=24
fcstlen=42

# access control
# hosts allowed to access data
[valid]

hosts=XXX.XXX.XXX.*
```

In this case, this is the directory where **avntrigger.sh** writes files extracted from text database. A special case is LLWSThread which relies on aircraft, radar and profiler data. The relevant directories are specified by keywords acars, radar and profiler respectively.

❶ Section [dis] specifies which ingest threads should run. For each item *tag* on the list there is a corresponding [dis_*tag*] section which defines the thread. Each section comprises several items listed below.

- name is one of predefined data sources which is specific to each module. Do not modify this value.
- source is the directory where AWIPS data files reside. This directory is monitored by **gamin** server daemon, gam_server.
- module specifies Python source code file. Do *not* change.
- nhours is a module specific parameter, in this case it is the number of hours of data that should be preserved in AvnFPS database.

❷ Section [dis_text] defines an alternate setup that bypasses AWIPS decoders and text database. TAFs and METARs are retrieved from files dumped by **acqserver** into a scratch directory. This configuration is not recommended for WFOs as it requires modification of baseline AWIPS configuration files, but you can always ask.

❸ Item [dis_text] defines thread processing reports retrieved from Postgres fxatext by **avntrigger.sh** via the trigger mechanism. This is delivered as the default WFO technique to get text products into AvnFPS nhours specifies the number of hours of data that should be kept in the AvnFPS database.

❹ ltg tag controls the processing of lightning observations. The tag distance defines the radius of circles centered at the latitude/longtitude pair given for the TAF sites. Lightning strikes within these circles will be extracted from AWIPS data files.

❺ [dis_rltg] tag defines a thread that processes LAMP thunderstorm guidance. lightning probability forecast which is currently available in the lower 48 states. OCONUS sites may safely remove this tag from the list in the [dis]section.

❻ [dis_llws] controls Low Level Wind Shear setup. The directories are the aircraft (ACARS) wind profiles, base radar file location (the site id and product type VWP will be appended by the thread). The profiler tag can be a comma-delimited list of directories. This allows LDAD directories as well as AWIPS ones to be listed.

❼ The [dis_file] describes a thread that reads the content of files in the directories listed on the source line. The base directory of each file is passed to the client, which can thus determine the data type. Currently this thread reads files produced by a GFE text formatter.

❽ The [dis_guid], specifies guidance data: MOS, LAMP and high-resolution point NAM-WRF model data. You may comment out some lines if a particular model guidance is not available in your area. nhours specifies the number of forecast hours shown in the various guidance viewers. At a minimum, 42 hours (the default) is shown up to a maximum 84 hours can be displayed.

❾ The access control section [valid] restricts access to AvnFPS server to client hosts with IP addresses matching those on the hosts line. A shell type match characters can be used: * matches any string, ? matches single character,

etc/px2finit.cfg

Specifies which servers should be started by **avnint** and their startup order and this file should not be changed.

```
# px2finit.cfg

[avnserver]

name=avnserver
order=0
wait=5

[avndrs]
name=avndrs
order=1
wait=3

[avndis]
name=avndis
order=2
wait=1

[avnxs]
name=avnxs
order=3
```

❶ For each server there is a separate section. The tag must be unique. A section consists of 2 or 3 items:

- name specifies server name
- order determines server startup sequence. Servers with lower order start first.
- wait is a wait time in seconds after a particular server is started (forked), before starting the next server on the list.

etc/gui.cfg

This file specifies settings and appearance of the AvnFPS Monitoring GUIs. Values set in this file is shared by all users.

```
# etc/gui.cfg
# specifies available monitor options for avnwatch

# Name servers
# Either local or ip address. Local means that broadcast is used to locate
# the server, which means the server must be on the same subnet
# The list of tafs is displayed in the startup (menu) GUI
[ns]
tags=local,

# colors in main GUI: 7 items
[colors]
tags=green3,grey,pale green,yellow,orange,red,purple


# editor tags
[editor_tags]
fatal=red
error=orange
warning=forest green

# miscellaneous features
[features]
tafeditor=1
twbeditor=0

# viewers: taf and metar must be present, taf must be first on the list
[viewers]
tags=taf,metar,gfsmos,gfslamp,tug,nammos,nam,grids

[viewer_taf]
module=TafViewer

[viewer_metar]
module=MetarViewer
label=Metars

[viewer_gfsmos]
module=MosViewer
label=GFS MOS
model=gfsmos

[viewer_gfslamp]
module=MosViewer
label=LAMP
model=gfslamp

[viewer_tug]
model = gfslamp
module = TUGViewer
label = TAF/LAMP

[viewer_nammos]
module=MosViewer
label=NAM MOS
model=etamos

[viewer_grids]
module=GridViewer
label=NDFD Grids

[viewer_nam]
module=EtaViewer
label= NAM DMO

[menus]
number=7

[menu_0]
items=metar

[menu_1]
items=persistence_1hr,persistence_2hr,persistence_3hr

[menu_2]
items=ltg

[menu_3]
items=rltg
```

```
[menu_4]
items=ccfp

[menu_5]
items=grids

[menu_6]
items=llws

[monitor_metar]

menu=Current Observation
module=MetarMonitor
items=tempo,wind,wx,vsby,sky
labels=tpo,wnd,wx,vis,cig
tempolevel=2
tempograceperiod=1800.0

[monitor_persistence_1hr]
menu=1hr Persistence
module=PersistMonitor
nhours=1

items=vsby,wind,wx,sky
labels=vis,wnd,wx,cig

[monitor_persistence_2hr]
menu=2hr Persistence
module=PersistMonitor
nhours=2
items=vsby,wind,wx,sky
labels=vis,wnd,wx,cig

[monitor_persistence_3hr]
menu=3hr Persistence
module=PersistMonitor
nhours=3
items=vsby,wind,wx,sky
labels=vis,wnd,wx,cig

[monitor_ltg]
menu=ltg
module=LtgMonitor
items=wx
labels=ts

[monitor_rltg]
menu=rltg
module=RadLtgMonitor
items=wx
labels=ts

[monitor_ccfp]
menu=ccfp
module=CCFPMonitor
items=wx
labels=ts

[monitor_grids]
menu=NDFD Grids
module=GridMonitor
from=2

to=6
items=vsby,wind,wx,sky
labels=vis,wnd,wx,sky

[monitor_llws]
menu=llws
module=LLWSMonitor
items=wind
labels=ws
```

1. The tags are a list of comma separated addresses of machines where the AvnFPS name server runs. Thus, the AvnFPS Monitoring GUI could, in theory, access different WFO servers provided the targeted WFOs allowed connections from external users.
2. Lists default colors used to indicate discrepancies between forecasts and observations/guidance, in increasing order of severity. The first 2 colors are used to indicate OK and Missing status, the remaining are used to mark severity codes. These are now "fallback" color scheme should the individual user not provide his/her own colors.
3. These are the colors used in the TAF text editor's syntax checker.
4. If set to 1, a TAF Editor button will be displayed in the main monitoring GUI below the menu bar.
6. Lists data viewers available in the TAF Editor window. For each tag listed, there must be a [viewer_tag]

section.

6. Each viewer requires some common parameters: the module attribute specifies a Python source code file; the label attribute defines the text displayed in the viewer window's tab. The MosViewer module requires one other additional value, model, to distinquish between difference guidance sources. You should not modify any of these attributes, with the exception of label.
7. Section [menus] specifies what is shown in the main monitoring GUI. For each item there must be a section [menu_#] listing attributes that will be monitored. Count starts from 0.
8. The items attribute is a list of comma separated tags which corresponds to a [monitor_item] section. The list generally consists of one item. However, if 2 or more items are on the list, the first item listed appears by default when the monitoring GUI is first shown.
9. Each monitoring function requires certain attributes: the module specifies a Python source code file; menu defines the text to be displayed at the top of the section; items is a list of monitored weather elements; the labels determine the text displayed in the indicators. To keep the GUI layout nicely aligned, keep the label names just 2 or 3 characters long. For the 'Current Observation' section, the tempolevel attribute is used to determine which monitoring rules are used when deciding if a TEMPO condition was not satisfied. Also the tempograceperiod attribute sets the minimum length of time which the TEMPO group condition is not realized before the forecaster is alerted.
10. nhours is the number of hours to look ahead assuming the observation does not change. Of course, the value of nhours should match the menu text.
11. from and to tokens define the interval of time, in hours, the gridded values are monitored against the TAF.

etc/wxplot.cfg

This is a configuration file for the weather plot GUI.

```
# wxplot.cfg

[print]                 ❶

cmd=convert - tmp/%s.jpg

[hours]                 ❷

back=24
forward=24

[vsby]                  ❸

bot=0.125
top=10.0

[cig]
bot=100
top=8000

[viewers]               ❹

tags=taf,metar,gfsmos,gfslamp,nammos,nam

[selected]              ❺

tags=taf,metar

[viewer_taf]            ❻

module=TafPlot
label=TAF
color=blue

[viewer_metar]
module=MetarPlot
label=METARs
color=red

[viewer_gfsmos]
module=MosPlot
label=GFS MOS
color=forest green
model=avn

[viewer_nammos]
module=MosPlot
label=NAM MOS
color=green
model=gfs

[viewer_nam]
module=EtaPlot
```

```
label=NAM DMO
color=brown

[viewer_gfslamp]
module=MosPlot
label=GFS LAMP
color=purple
model=gfslamp
```

❶ Command to be executed when the **Print** is pressed. A window dump file (xwd format) is passed via standard input stream.
❷ Number of hours of data to display in the past and future based on the current hour.
❸ Ceiling and visibility plot limits at the top and bottom of the graphics.
❹ This section lists data viewers available in the Weather Plot window. For each tag listed, there must be a [viewer_tag] section.
❺ Data viewer(s) enabled by default when GUI is first displayed.
❻ Each viewer requires certain parameters. 'module' tag specifies a Python source code file, 'label' tag defines text displayed as a toggle button name. Some modules require additional values. You should not modify any of those values, with the exception of the labels and colors.

etc/xmit.cfg

This configuration file is used by **avnxs** which sends forecast and verification products out for dissemination.

```
# etc/xmit.cfg

[verification]          ❶

wmo = NXUS98 KPIT
awips = KPBZVFTPBZ
fcstid = 0
period = 6

[transmission]          ❷

# how often checks for new files (seconds)
frequency = 15
# reject older than (hours)
old = 3
```

❶ This section defines parameters used by the program to transmit the NXUS98 verification product to NWS headquarters. wmo and awips are the WMO and AWIPS headers. These headers must exist in the file `/awips/fxa/data/afos2awips.txt`. The transmission file name contains typically unused forecaster id 0 (zero). The verification product is transmitted every 6 hours by default.
❷ You may want to adjust the attribute 'old' value. It defines the length of time (in hours) a product may reside in the pending queue for transmission, before moving it to the 'bad' directory.

etc/triggerTemplate

Made by **avnsetup**'s Trigger Editor. It should not be necessary to edit it.

2.2. TAF configuration files

The TAF configuration files reside in directory `etc/tafs`. For each TAF site CCCC there is a subdirectory `etc/tafs/CCCC` containing template files, one per each forecast period and site info data `etc/tafs/CCCC/info.cfg`.

Site Info File

This file is normally created by the Taf Site Editor GUI, normally invoked from **avnsetup**.

```
# etc/tafs/KIAD/info.cfg

[headers]
wmo = FTUS41 KIAD
afos = WBCTAFIAD

[thresholds]

cig = 200,600,1000,2000,3100
radar_cutoff = 100,0
vsby = 0.5,1.0,2.0,3.0,6.0
profiler_cutoff =
tafduration = 24

[sites]

radars = KLWX,TIAD
gfsmos = KIAD
gfslamp = KIAD
metar = KIAD
eta = KIAD
ngmmos = IAD
profilers =
etamos = KIAD
acars = IAD

[geography]

lat = 38.96
runway = 180,110
lon = -77.45
elev = 98

[qc]
impact = 1

climate = 1
currentwx = 1
```

1. Section [thresholds] defines ceiling and visibility categories used by the monitoring system. Cutoff values correspond to heights *in meters* below which data will be ignored. As in the example above, for the KLWX radar, any VWP data above 100 meters to 610 (2000 ft) meters will be used in the wind-shear calculations. This feature is useful if your VWP source(s) is (are) routinely contaminated by non-meterological phenomenon at the lowest levels near the site. ACARS data has been QC and are used in their entirety: there are no "cutoff" value for it. The length of the aerodrome forecast is given, in hours, by the tafduration attribute.
2. Section [sites] defines IDs used by other data sources. Some TAF sites may have more than one associated observation. In that case, separate the ids by a comma.
3. Section [geography] provides location of the airport. elev is in *meters*, runway directions in whole degrees *true north not magnetic north* and are used to calculate cross/head/tail winds.
4. Section [qc] creates the default set of QC checks for this particular airport. Each check can be toggled on/off as appropriate and desired. Once saved, these checks will be performed on the TAF for this airport location when using the left-most mouse click on the QC button in TAF Editor.

Monitoring Rules

The default configuration files for all monitors are located in the directory `etc/tafs/XXXX`. Currently these are `grids.cfg`, `ltg.cfg`, `mtrs.cfg`, `rltg.cfg`, `ccfp.cfg`, and `llws.cfg`. These files serve as a fallback or default monitoring behavior for TAFs that do not their own version of one or more of these files in their `etc/tafs/<TAFID>` directory.

All these files can be edited with the "Montoring Rules" GUI via **avnsetup**. Here is an example of rule definition file for monitoring of METARs:

```
# etc/tafs/XXXX/mtrs.cfg

[rule_11]
                    ❶
severity = 5
msg = Visibility difference of 3 or more categories
unique = 1
type = vsby
method = VsbyCatDelta
ncat = 3
.......
[rules]
                    ❷
active = 0,1,2,3,4,5,6,7,8,9,10,11
.......
[rule_1]
severity = 3
msg = Freezing precipitation in TAF and not in obs
unique = 0
type = wx
method = WxTafDelta
wx = FZRA,FZDZ,PL
.......
```

❶ Each [rule_*N*] provides arguments to the monitoring function, specified by the method keyword. The following tags must exist for each rule:

1. severity: a number in the range 1 through 5 corresponding to colors defined by the forecaster or in a fallback condition, in `gui.cfg`, see ❷ [14].
2. msg: balloon message displayed in the monitoring GUI
3. type: one of cat, sky, vsby, wind, wx
4. unique: 0 or 1.

Each function may accept additional arguments, such as ncat. In this case ncat is the number of categories the visibilities in a TAF/METAR pair must differ before the rule becomes true. You may and should create several rules using the same method but with different values for ncat and severity.

❷ Section [rules] defines the list of rules to monitor. For each number there must be a corresponding [rule_*N*] section.

If you want to have rules dependent on the TAF site, copy the relevant file to the TAF site directory and modify it there. The monitoring program first tries to find the files in the site directory, if it can't find it there, uses the one from `etc/tafs/XXXX`.

Important

There are no default hardcoded rules, the files in `etc/tafs/XXXX` must exist.

Impact QC Definition File

With the Impact QC feature, the forecaster can be alerted when a prepared TAF has a significant impact on airport operations prior to issuing the product. This file is optional and cannot be modified with **avnsetup**, you must use a text editor instead. Shown below is a configuration file with additional rules added that are specific to Pittsburgh International Airport, `etc/tafs/KPIT/impact.cfg`:

```
# impact.cfg

[conditions]
                    ❶
items=cond_1,cond_2,cond_3,cond_4,cond_5
```

```
[cond_1]                    ❷

tag=FC
level=1
text=Impact: Fuel-Alternate vsby<3 or cig<2000
expr=vsby<2.95 or cig<2000

[cond_2]
tag=FC
level=2
text=Impact: LIFR conditions vsby<1 or cig<500
expr=vsby<0.95 or cig<500

[cond_3]
tag=ws
level=1
text=Wind shift vs previous group resulting in >6kt change along PIT runway 28R/10L,may affect runway config
expr=wind[0].shift and wind[0].runway>6

[cond_4]
tag=xw
level=1
text=Cross wind component >14KT on PIT runway 28R/10L
expr=wind[0].cross>14

[cond_5]
tag=xw
level=2
text=Cross wind component >24KT on PIT runway 28R/10L
expr=wind[0].cross>24
```

❶ Section [conditions] defines the list of conditions to check. For each tag [cond_*N*] there must be a corresponding section with that name.

❷ Each [cond_*N*] provides arguments to the impact check function. The check is done for each TAF group. The following tags must exist for each rule:

1. expr: a Python expression to be evaluated. This expression can contain following variables: vsby, cig, wind, ts. vsby is in miles, ceiling in feet. wind is an array that has length equal to the number of runways. Each element of this array contains 3 values: runway, cross and shift. The wind vector is represented in a Cartesian coordinates along and perpendicular to the runway. The absolute values of these components are wind[*N*].runway and wind[*N*].cross, where *N* is runway number, counted from 0, as listed in the site configuration file, see section : “Site Info File” (page 18). wind[*N*].shift is a Boolean value, indicating that the runway component changed sign from one TAF period to another. ts is a Boolean value denoting whether thunderstorms are present in the TAF.

2. text: message to be displayed when the expression evaluates to True.

3. level: highlight background color, 1 - green, 2 - orange, 3 - red.

4. tag: used to group messages, so only the message with the highest level is displayed. In this example, if visibility is 1/2SM, only the LIFR message is shown in a popup window of the TAF Editor.

Climate QC Definition File

The file `etc/tafs/XXXX/climqc.cfg` is used to define category thresholds used while creating climate QC count files. Currently this file cannot be modified with **avnsetup**, you must use a text editor. An example:

```
#etc/tafs/XXXX/climqc.cfg

[args]          ❶

alpha=0.3
showdetails=0

[thresholds]            ❷

vsby=0.27,1.1,2.9,6.1
cig=205,605,1005,3105,40000
ff=4.1,10.1,20.1
dd=22.5,67.5,112.5,157.5,202.5,247.5,292.5,337.5
```

❶ The parameter alpha (see Appendix C: "*Climate Quality Control*" []) is used to determine likelihood of occurence of a combination of weather elements in the forecast based on conditional probabilities. Alpha should be a positve number, less than one. The showdetails flag (0=no;1=yes) turns detailed statistics display.

❷ The thresholds determine category ranges. Each of arguments is a list of values $v_1 < v_2 < ... < v_n$, with implicit $v_1 = 0$ and $v_{n+1} = \#$. Category `k` consists of all values `v` such that $v_k \# v < v_{k+1}$.

The value 40 000 ft on the ceiling list serves to distinguish between limited and unlimited ceiling, which is coded as 99 998 ft.

If you want to have thresholds dependent upon the unique operational constraints at the specific TAF site, copy this file to the TAF site directory and modify it there. The program **avnqcstats** which creates the count files first tries to find the file in the site directory, if it can't find it there, uses the one from `etc/tafs/XXXX` directory.

TAF Product Definition Files

Located in `etc/tafs`. Normally created by the Taf Product GUI invoked from **avnsetup**. An example file:

```
# etc/tafs/PBZ_TAFS.cfg

# list of TAF sites
[sites]
workpil=PITWRKTAF
idents=KPIT,KAGC,KBVI,KLBE,KHLG,KMGW,KZZV,KFKL,KDUJ
collective=
```

Default Product File

The optional `DEFAULT` file contains name of the configuration file, without the `.cfg` extension. that should be loaded on startup of the monitoring GUI. If the `DEFAULT` is not present, the first TAF product configuration file in alphabetical order is used by the AvnFPS clients.

TAF Formatter Configuration Files

The TAF Formatter requires three configuration files. These files are located in the directory `etc/tafs/XXXX`. With AvnFPSOB9.2, `flt_cat.cfg` and `grp_taf.cfg` files can now be customized for each TAF site and moved to its respective `etc/taf/CCCC` directory.

File `flt_cat.cfg` contains the threshhold values of visibility and ceiling height that are used to determine aviation flight categories. The default values are identical to those provided in NWS Instruction 10-813. The thresholds can be changed to site-specific values.

```
#etc/tafs/XXXX/flt_cat.cfg
[cig]
thresholds=200,500,1000,3000,3500,4000,5000,8000,10000,12000,15000,18000,20000
[vis]
thresholds=0.5,1,3,5 ❶
```

❶ These thresholds divide the spectrum of ceiling and visibility into categories with an implied 0 at the beginning and 'infinity' at the end of each list. Thus as defined here, ceiling category 1 corresponds to cloud heights from 0 to 200 feet, category 2 from 200 to 500 feet, etc. Similarily for visibility, with category 1 corresponding to visibilities LTE to 1/2 mile. Category 5 corresponds to visibility greater than 5 miles.

File `grp_taf.cfg` contains the configurable threshold values that are used to form concise TAFs. Detailed description for each category has been provided inside the file.

```
#
# This file contains thresholds that control the algorithm that converts
# numerical/tabular gudiance into 'guidance' TAFs.
#
# Probability thresholds used for MOS.  This section also acts as a default
# section if 'gfslamp' and 'grid' sections are not provided
```

```
#
[mos]
#
# Probabilities (POT and POP06) equal and above which precipitation and/or
# thunderstorms will be included in the prevailing group.
#
prev_precip=80
prev_tstm=80
#
# Probabilities (POT and POP06) equal above which the tempo group is formed
# containing precipitation and/or thunderstorms.
#
tempo_precip=36
tempo_tstm=36
#
# Probabilities (POT and POP06) equal and above which the prob30 group is formed
# containing precipitation and/or thunderstorms.
#
prob30_precip=26
prob30_tstm=26
#
#Same as above but used after the 9th hour in forecast
#
prev_precip_af_9hr=90
prev_tstm_af_9hr=90
tempo_precip_af_9hr=46
tempo_tstm_af_9hr=36
#
#
[gfslamp]
#
# A new set of thresholds for LAMP due to the increased forecast resolutions:
# hourly probability for precipitation and two-hour probability for thunder
#
prev_precip=50
prev_tstm=35
tempo_precip=36
tempo_tstm=26
prob30_precip=25
prob30_tstm=15
#
# Same set of thresholds used for after the 9th hour in forecast
prev_precip_af_9hr=60
prev_tstm_af_9hr=45
tempo_precip_af_9hr=46
tempo_tstm_af_9hr=36
#
# Boolean (yes/no) if PROB30 groups are wanted in guidance TAFs
[prob30]
value=yes
#
[def_singleton]
#
#
# The longest time span in hours that makes this group combinable with
# a longer, adjacent (in time) group. If the duration of a group is longer
# than this value, this group stays alone in the final TAF.
#
short_dt=1 ❶
#
# The shortest time span in hours that allows its shorter duration
# neighbor to be combined with it.
#
long_dt=15 ❷
#
# Maximum category difference between the long and the short duration groups. If the
# difference in flight category between the two groups is greater than this value,
# they are not combined.
#
# Categories are treated separately for visibility and ceilings.
#
# To see the breakpoints for the ceiling and visibility categories, see
# etc/tafs/XXXX/flt_cat.cfg (flight category) configuration file.
#
# First the visibility
#
dc_vis=1 ❸
#
# Lowest visibility category to which the combining algorithm is applied
#
low_c_vis=5 ❹
#
# Highest visibility category to which the combining algorithm is applied
#
high_c_vis=5 ❺
#
# Now, the ceiling . . .
```

```
#
dc_cig=1 ❻
#
# The lowest ceiling category to which the combining is applied
#
low_c_cig=13 ❼
#
# Highest ceiling category to which the combining is applied
high_c_cig=14 ❽
#
# Earliest forecast time in hrs to which the combining algorithm can be applied
low_p=12 ❾
#
# Maximum allowable ceiling height in hundreds of feet when thunderstorms
# are forecasted
#
[cb_hi]
value=50
#
# Maximum speed which is considered to be not a calm wind.
[wind]
calmwd=2
#
# Wind speed and direction change in wind speed above which combining will not be applied.
high_wind=10
wind_chg=4 ⓫
#
# Boolean (yes/no) flag if dry thunder is allowed in TAF
[dryts]
value=yes
#
# Boolean (yes/no) flage if CB without thunder is allowed in TAF
[cbwots]
value=no
```

Note

Items 1 through 9 control how two *dissimilar* (in visibility and/or ceiling) groups, one 'short', the other 'long', can be further combined.

❶ The maximum duration of the smaller group that can be combined with the larger, dissimilar one. When combined, the larger group values are not adjusted by the smaller group's values.
❷ The minimum length of time to be considered the 'large' group candidate for combining with smaller ones.
❸ The largest visibility category *difference* allowed between TAF groups and still be able to combine.
❹ The lowest visibility category allowed to be considered for combining. This value is related to the values assigned to the vis thresholds in `flt_cat.cfg` file.
❺ The highest visibility category allowed to be considered for combining. This value is related to the values assigned to the vis thresholds in `flt_cat.cfg` file. Usually set to the number of visibility 'breakpoints' in vis thresholds.
❻ The largest ceiling category *difference* allowed between TAF groups and still be able to combine.
❼ The lowest ceiling category allowed to be considered for combining. This value is related to the values assigned to the cig thresholds in `flt_cat.cfg` file.
❽ The highest ceiling category allowed to be considered for combining. This value is related to the values assigned to the cig thresholds in `flt_cat.cfg` file. Usually set to the number of ceiling 'breakpoints' in cig thresholds.
❾ Number of hours when the combining algorithm will be applied to groups valid beyond the TAF's beginning valid time.
⓫ Maximum allowable difference in speed (knots) and direction (tens of degrees) between groups allowed to be combined.

Finally, `grid_probs.cfg` contains configuration data on how the probability attributes on IFPS grids are translated into numerical probability values that will be used to form TAFs. Remember that precipation is only included in the TAFs when certain probability threshold is exceeded. This file cannot be customized to individual TAF sites.

```
#tables used to convert prob symbols in grids to a pop value
```

```
[before9hr]
S=30
IS=30
WS=30

SC=50
O=50
C=50

D=70
WP=70
NM=70
L=70

[after9hr]
S=20
IS=20
WS=20

SC=30
O=30
C=30

D=50
WP=50
NM=50
L=50
```

2.3. X resources configuration files

The default and individual resource files are located in the directory `etc/app-resources`. To edit the default file `etc/app-resources/X` you can use any text editor, however, this shouldn't be necessary. Individual files have naming convention `X.N`, where N is the forecaster number from `etc/forecasters`. These files are created by a graphical editor available from the AvnWatch GUI menu.

Here is the listing of the default file `etc/app-resources/X`

```
!! default X resources configuration file for AvnWatch GUI, TAF Viewer, and Editor
!! default font
!! change 100 to 140 if you want bigger default font
*font:                          -adobe-helvetica-bold-r-normal-*-*-100-*-*-p-*-iso8859-9
!! default colors
*background:                    grey70
*foreground:                    black
!! text window font
*Text.font:                     7x14
!! text window colors
*Text.background:               #2f3f6f
*Text.foreground:               #ffffff
!! other text window specific options
*Text.width:                    74
*Text.height:                   24
*Text.cursor:                   hand1
!! forecast editor font - may want bigger
*textEditor.font:               8x16
!! forecast editor colors - may want different
*textEditor.background:         #1f2f6f
*textEditor.foreground:         white
*textEditor.width:              70
*textEditor.height:             12
!! cursor width and color
*textEditor.cursor:             arrow
*textEditor.insertWidth:        5
*insertBackground:              yellow
!! text viewer window options
*textViewer.width:              74
*textViewer.height:             12
!! forecast editor orientation: horizontal or vertical
*orientation:                   vertical
!! listbox font
*Listbox.font:                  7x14
!! entry font
*Entry.font:                    7x14
!! entry window colors
*Entry.background:              white
*MessageBar*Entry.background:   grey85
!! balloon message (popup) font
*Balloon.Label.font:            7x14
!! balloon message alert placement
*impactPlacement:         top
!! modifies behavior of dialogs
*transientDialogs:              1
!! confirmation on closing editor
*confirmClose:                  0
!! confirmation on sending amendments/corrections
*confirmSend:                   1
!! check Transmisson Server Pending Queue and alert user if forecast will be removed.
*checkPendingQueue:     1
!! status alert colors: colors shown on the 'christmas tree'
*alertLevel0:     green3
*alertLevel1:     grey
*alertLevel2:     pale green
*alertLevel3:     yellow
*alertLevel4:     orange
*alertLevel5:     red
*alertLevel6:     purple
!! alert options: use colors for the first level that should result in
!! notification - pale green, yellow, orange, red, purple or disabled
*notifyDeiconify:               alertLevel3
*notifyRaise:                   alertLevel4
*notifyPlay:                    alertLevel5
!! not implemented
!!*notifyTalk:                  disabled
*playFile:                      /awips/fxa/data/sounds/asterisk.au
!! blink on new notification
*blink:                         1
!! warning/error level to disallow send:
!! 'always', 'warning, 'error', 'fatal'
*disallowSend:                  error
!! forecast editor options
!! use template, one of: template, merge, latest
*loadOrder:                     merge
!! ask for transmission time when sending routine forecasts
```

```
*asktime:                        1
!! periodically save bulletins in a backup file
*autosave:                       1
!! Save to disk and text database when Save button is pushed
*alwaySaveToDB:                  0
!! update issue and valid times on QC
*updatetimes:                    1
!! print forecasts on send
*autoprint:                      0
!! Insert/Overwrite
*insert:                         1
!! Word-wrap long lines: either none or word
*wrap:                           word
!! Amd, Rtd, Cor buttons on the right
*amdbuttons:                     1
!! number of TAFs to display
*numTafs:                        1
!! number of hours of METARs to display
*numHours:                       6
!! number of hours of METARs to display when all stations are to be displayed in viewer
*numHoursAll:                 1
!! show bulletin headers in the text window
*showHeaders:                    1
!! show decoded METARs
*showDecoded:                    1
!! show category probabilities for MOS
*showProbs:                      1
!! show guidance TAFs as formatted for the next routine issuance
*showRoutine:                 1
!! Format of MOS and Grids reports: raw, long or short
*showFormatted:                  short
!! flight category highlights
*highlightFlightCat:             0
*lifrColor:                      #6c1916
*ifrColor:                       #5b3c2c
*mvfrColor:                      #2c484c
*vfrColor:                       #112346
!!
!! editable by knowledgable people only
!! prevents line wrap in message boxes
*wrapLength:                     0
```

Lines starting with "!!" are comments. Resource names begin with a "*" and end with ":". Anything after a ":" is a resource value. Many of the allowable values are not well documented, or cryptic. For example, valid values for cursor shape in a text window

```
*Text.cursor:                    hand1
```

are defined in file `/usr/include/X11/cursorfont.h`. You have to know that the leading XC_ in line

```
#define XC_hand1 58
```

must be stripped and 58 ignored. Valid color names are in `/usr/lib/X11/rgb.txt`, but if you don't find one you like, a hexadecimal number can be used instead, such as

```
*lifrColor:                      #6c1916
```

We suggest that you use the graphical resource editor available from the in the monitoring GUI, a.k.a., **AvnWatch** Section 2.4: “Menu Bar” (page73) toolbar to make changes to your configuration.

3. Starting and Stopping AvnFPS

All AvnFPS servers, clients and utilities are started with a shell script **avnstart.sh**. This script is responsible for setting appropriate environmental variables required by AvnFPS. The variables are actually set in the file `bin/avnenv.sh` which is sourced by **avnstart.sh**. The first argument is program name (a Python script residing in the directory `py` *without* the extension `.py`). Remaining arguments, if any, are passed to the program without modification.

- To start **avnsetup** from the command line, enter:

```
lx1-wfo: /awips/adapt/avnfps/bin/avnstart.sh avnsetup
```

- To start **avnmenu** from the command line, enter:

```
lx1-wfo: /awips/adapt/avnfps/bin/avnstart.sh avnmenu
```

3.1. Starting AvnFPS Servers

AvnFPS requires four servers. These are:

- **avnserver**: Helps client processes find the services they need (name service) and notifies clients when data are available (event service).
- **avndis**: Data Ingest Server. Copies data from various AWIPS sources into AvnFPS directory tree, reformatting as needed.
- **avndrs**: Data Request Server. Answers requests for data from client processes.
- **avnxs**: Forecast Transmission Server. Manages transmit queues and the interface to the AWIPS software that transmits products.

The order in which the servers are started is important. **avnserver** must be started first. There can be only one instance of avnserver within WFO subnet, otherwise resource and data conflicts will result.

avndis depends on **avndrs** to process Low Level Wind Shear data, so **avndrs** needs to be started next. The simplest way to avoid these problems is to not launch these servers directly. The application **avninit** searches for active instances of all four AvnFPS servers and launches the ones that are not running. This application is modelled on the Unix™ SYSV **init** and the AWIPS **DataController**. Like **init** and **DataController**, **avninit** is a persistent process.

Note

avninit will attempt to restart a failed server 10 times within an hour. When the number of restarts exceeds 10, **avninit** will stop trying to restart the failed server.

avninit is invoked by the AWIPS startup script **px2apps** together with other AWIPS applications. If you have to start the servers afterwards, use the shell script **remoteServers.sh**. The script accepts the following command line arguments:
`remoteServers.sh` [start | stop | restart]

This script has to be run as user fxa. It uses **ssh** to start/stop **avninit** on `px2`. Therefore it can be run from any workstation. It is located in the directory `/awips/adapt/avnfps/bin`. Example usage:

```
lx2-wfo: /awips/adapt/avnfps/bin/remoteServers.sh start
```

To list avnfps processes use the **ps** command. A typical output on `px2` is:

```
px2-wfo: ps -ejHf | grep avnpython
fxa   18202     1 18201 18201  0 Apr28 ? 00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avninit.py px2f
fxa   18203 18202 18201 18201  0 Apr28 ? 00:06:46  avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n
px2f
fxa   18224 18202 18201 18201  0 Apr28 ? 00:02:10  avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18239 18202 18201 18201  0 Apr28 ? 00:17:02  avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18250 18202 18201 18201  0 Apr28 ? 00:00:55  avnpython /awips/adapt/avnfps/OB9.2/py/avnxs.py -d -n px2f
```

If you wish to see all the threads in hierarchical format, add -L option to **ps**:

```
px2-wfo: ps -eLf | grep avnpython
fxa   18202     1 18202  0  1 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avninit.py px2f
fxa   18203 18202 18203  0 19 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18204  0 19 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18222  0 19 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18223  0 19 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18229  0 19 Apr28 ?  00:00:26 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18257  0 19 Apr28 ?  00:00:53 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18258  0 19 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
```

```
fxa   18203 18202 18260  0 19 Apr28 ?  00:01:04 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18262  0 19 Apr28 ?  00:01:10 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 18267  0 19 Apr28 ?  00:00:31 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 12155  0 19 May01 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 12156  0 19 May01 ?  00:00:09 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 12158  0 19 May01 ?  00:00:37 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 21876  0 19 06:09 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 21877  0 19 06:09 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202 21878  0 19 06:09 ?  00:00:02 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202  9029  0 19 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202  9030  0 19 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18203 18202  9032  0 19 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n px2f
fxa   18224 18202 18224  0 23 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 18226  0 23 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 18227  0 23 Apr28 ?  00:00:04 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 18272  0 23 Apr28 ?  00:00:08 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 12157  0 23 May01 ?  00:00:24 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 12159  0 23 May01 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 12160  0 23 May01 ?  00:00:01 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 12161  0 23 May01 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 12162  0 23 May01 ?  00:00:04 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 12186  0 23 May01 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 16203  0 23 May01 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 21879  0 23 06:09 ?  00:00:01 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 21880  0 23 06:09 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 21881  0 23 06:09 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 21882  0 23 06:09 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 21883  0 23 06:09 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 23108  0 23 11:12 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202 27969  0 23 11:23 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202  9031  0 23 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202  9033  0 23 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202  9034  0 23 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202  9035  0 23 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18224 18202  9036  0 23 11:58 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n px2f
fxa   18239 18202 18239  0 10 Apr28 ?  00:00:01 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18252  0 10 Apr28 ?  00:00:25 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18253  0 10 Apr28 ?  00:00:23 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18254  0 10 Apr28 ?  00:06:03 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18255  0 10 Apr28 ?  00:00:03 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18256  0 10 Apr28 ?  00:00:01 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18259  0 10 Apr28 ?  00:00:00 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18268  0 10 Apr28 ?  00:01:22 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18271  0 10 Apr28 ?  00:04:27 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18239 18202 18273  0 10 Apr28 ?  00:03:42 avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n px2f
fxa   18250 18202 18250  0  1 Apr28 ?  00:00:55 avnpython /awips/adapt/avnfps/OB9.2/py/avnxs.py -d -n px2f
```

Note

The -H (process hierarchy) option cannot be used with the thread (m, L or T) request.

3.2. Stopping AvnFPS Servers

Normally you should use the script **remoteServers.sh** to stop AvnFPS servers:

```
lx2-wfo: /awips/adapt/avnfps/bin/remoteServers.sh stop
```

Another option is to log on to the host(s) where AvnFPS servers run and use the **avnkill** command. This command is also located in `/awips/adapt/avnfps/bin` directory. The **avnkill** first issues a `SIGTERM` signal, then it waits 30 seconds for all threads to terminate gracefully. If the threads do not exit, **avnkill** issues a `SIGKILL` signal to those processes that continue to run. Yet another option is to send a `SIGTERM` signal to **avninit**. For example, the command to do this would be:

```
px2-wfo: /usr/bin/pkill -f avninit -u fxa
```

This should stop all the servers since **avninit** sends `SIGTERM` to all its child processes before quitting.

Note

When **avninit** stops, for whatever reason, the existing servers are stopped as well.

Under special circumstances, (e.g., a change to a configuration file), you may want to stop and start a single server. Since **avninit** monitors the servers and starts those which are not running, the only action needed is to kill the process in question. Bear in mind that the servers are threaded and only the top-level thread accepts signals.

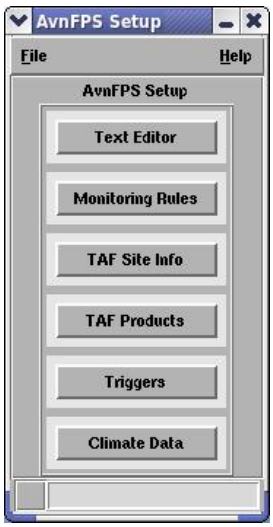
For the **ps** listing in the previous section, the hierarchy is represented by order. For instance, to restart the **avndis** server, type:

```
px2-wfo: /usr/bin/pkill -f avndis -u fxa
```

This will kill the AvnFPS data ingest server process and its threads. Once the **avndis** server quits, **avninit** will start a new **avndis** server instance which will then read the changed configuration file.

4. Avnsetup

avnsetup is a graphical user interface (GUI) which can be used to configure many features of AvnFPS. In most cases, the data displayed and modified in **avnsetup** are contained in the various files described in Section 2, “Configuring AvnFPS” [10].



avnsetup main GUI

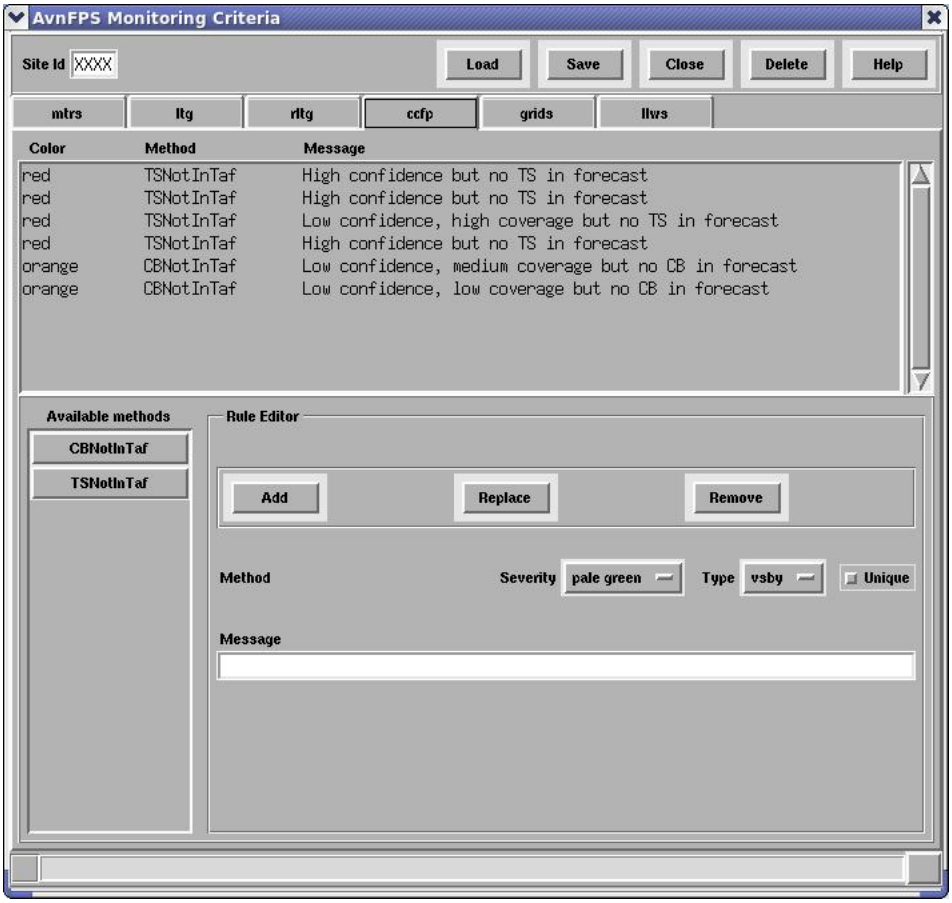
This section describes tasks that can be accomplished with this program.

Important

The AvnFPS severs and, often, **AvnWatch** GUI are required to be restarted for changes made using the following setup GUIs, except the Trigger Editor, to take affect. The reason for this, is that the underlying files, Section 2, “Configuring AvnFPS” [10], are read only once into memory on startup of the servers and **AvnWatch**.

4.1. Editing Monitoring Rules

From the **avnsetup** main GUI, select the **Monitoring Rules** button. The Monitoring Criteria Editor will be shown.



Monitoring Criteria Editor

The Monitoring Criteria Editor contains several pages, one for each monitored data source. Pages are selected via “recipe tabs” near the top of the menu. The labels on the tabs refer to the following data sources:

- Rules for monitoring METARs (mtrs)
- Rules for monitoring lightning observations (ltg)
- Rules for monitoring lightning nowcasts (rltg)
- Rules for monitoring Colaborative Convective Forecast Product (ccfp)
- Rules for monitoring forecast grids (grids)
- Rules for monitoring low level wind shear (llws)

Each page has an identical layout: the top part lists currently defined rules, the bottom part contains a list of all available methods and the area where you can view, modify, add or delete a rule.

AvnFPS supports site-specific monitoring rules as well as a set of default rules. When the rule editor is started, the

default rules are loaded. This is indicated by XXXX in the **Site Id** entry field.

To load the default set of rules:

- Enter the 'XXXX' into the box labeled **Site Id**.
- Press **Enter** or select the **Load** button.

To load the specific rules for a site:

- Enter the Station ID into the box labeled **Site Id**.
- Press **Enter** or select the **Load** button.

To create a new set of site-specific rules:

- Modify the rules using the instructions below.
- Enter the applicable Station ID into the box labeled **Site Id**.
- Select the **Save** button.

To remove site-specific rules:

- Enter the Station ID into the box labeled **Site Id**.
- Select the “recipe tab” for the data source for which you want to remove the rules.
- Select the **Delete** button.

The lower half of the Monitoring Criteria Editor allows you to view and modify each of the monitoring rules. The current set of rules is listed by color and message in a selectable list. The Rule Editor at the bottom of the menu shows details of the rule that is currently selected. To the left to the Rule Editor is a list of Available Methods. These methods are software techniques that compare the current forecast to the latest values received from the data source. Each method has its own set of arguments that configure its behavior.

To view the details of an existing rule:

- Select the rule from the list.

To modify an existing rule:

- Select the rule from the list.
- Use the Rule Editor to make the necessary changes. See Appendix B, *Monitoring Rules* [113] for a description of editor options.
- Select **Replace**.
- Select **Save** to write changes to the configuration file.

To remove a rule:

- Select the rule from the list.
- Select **Remove**.

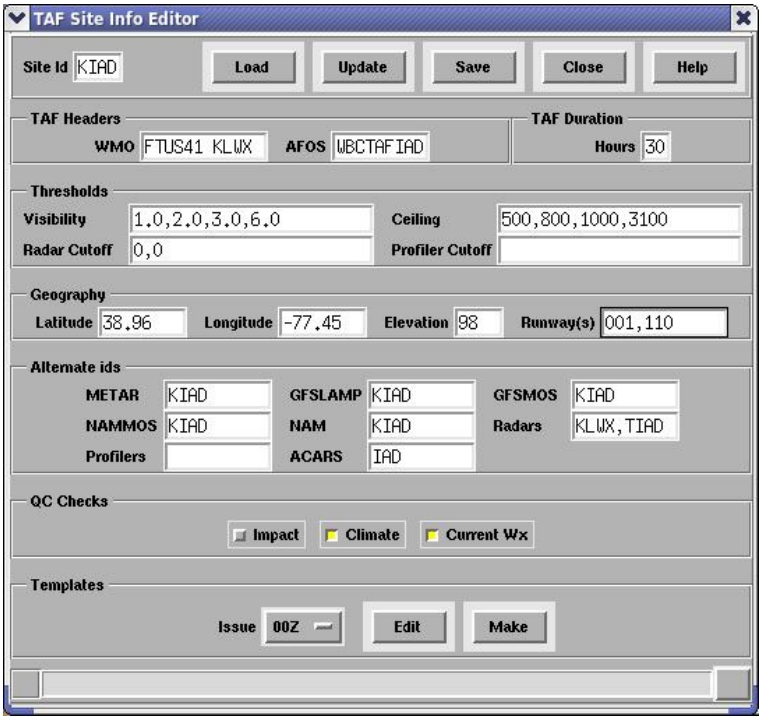
- Select **Save** to write changes to the configuration file.

To add a new rule:

- Select a method from the "Available Methods" list. Appendix B, *Monitoring Rules* [113] provides detailed description of each one. The Rule Editor will update with the appropriate fields. You can see method details by pointing the cursor to the respective button on the list.
- Use the Rule Editor to modify arguments as needed. Entering a text message is no longer necessary, if left blank, AvnFPS will generate one for you.
- Select **Add**.
- Select **Save** to write changes to the configuration file.

4.2. Editing TAF Site Information

From the **avnsetup** main GUI, select the **TAF SiteInfo** button. The TAF Site Info Editor will be shown.



TAF Site Info Editor

The TAF Site Info Editor is an entry form for data specific for a particular TAF site. Most of the fields can be initialized from data residing in AWIPS files or from built-in values. You must enter data for all sites before creating a product definition (that is, a "collective").

Site Id	TAF site ID
TAF WMO TAF AFOS	WMO header used to transmit forecast PIL used during transmission
TAF Duration	Length, in hours, of the regularly issued TAF.
Visibility Ceiling	A comma separated list of visibilities (miles) defining categories used by the monitoring modules. Given the list $0 < V_1 < V_2 < ... < V_n$, the category k consists of visibilities V such that $V_{k-1} <= V < V_k$, k = 1,..., n+1 and $V_0 = 0$, $V_{n+1} = \#$. A comma separated list of ceilings (feet) defining categories used by the monitoring modules. The categories are defined the same manner as above.
Radar Cutoff	A comma separated list of heights in meters defining lowest height to be used in computing LLWS using this radar. The length of the cutoff list must be equal to the number of radars listed in the 'Radar' text field below.
Profiler Cutoff	A comma separated list of heights in meters defining lowest height to be used in computing LLWS using this profiler. The length of list must be equal to the number of profilers listed in

	the 'Profiler' text field below.
Latitude	Site latitude, in degrees north
Longitude	Site longitude. (west longitudes are negative)
Elevation	Site elevation, in meters
Runway(s)	A comma separated list of runway directions in degrees. Used to calculate cross-, head- and tail-wind components.
METAR	METAR site id associated with the TAF, most likely the same as TAF.
NAM	The closest site id found in NAM BUFR message associated with the TAF.
GFSMOS	The closest site id found in GFS MOS message associated with the TAF.
NAMMOS	The closest site id found in NAM MOS message associated with the TAF.
GFS LAMP	The closest site id found in GFS LAMP message associated with the TAF, usually the same as GFS MOS identifier.
Radars	A comma separated list of radars associated with the TAF. Can be empty.
Profilers	A comma separated list of profilers associated with the TAF. Can be empty.
Impact	When selected Impact Check will be performed when QC button is pushed
Climate	When selected Climate Check will be performed when QC button is pushed
Current Wx	When selected Current Weather Check will be performed when QC button is pushed

Note

Airport runway directions, as entered in this editor, are based on *true north*, not magnetic north. For purposes of computing crosswinds, only one runway heading is required as it does not matter from the pilot's perspective trying to land his/her aircraft whether the crosswind goes from left-to-right or right-to-left. It's the magnitude of the crosswind that's important.

If head- or tail-winds are important for monitoring, then the airport runway heading must be carefully chosen. One should consult Appendix-B LLMetar rule's description for guidance.

To create data for a new TAF site:

- Enter the Station ID into the box labeled **Site Id**.
- Select the **Update** button. Many of the fields will be filled in. The headers are read from AWIPS file `afos2awips.txt`. Latitude, longitude and elevation (in meters) come from `metarStationInfo.txt`. Other values are built-in defaults.
- Modify displayed entries as needed. The **METAR** text field must be filled in, all others fields in the 'Alternate ids' are optional. A simple validation test is performed when you type. The program will not allow you to enter obviously incorrect values, incomplete entries are indicated by a pink background.
- Select the **Save** button.
- Create templates as described in the next section.

To modify TAF site data:

- Enter the Station ID into the box labeled **Site Id**.
- Press the **Load** button.

- Modify displayed entries as needed.
- Select the **Save** button.

4.3. Editing TAF Site Templates

The bottom part of TAF Site Editor provides tools to create and edit template files that can be used to initialize forecasts. There is a separate template for each forecast issue time. This allows to enter issue specific phrases such as `AMD NOT SKED AFT ....`

To create TAF templates:

- Enter the Station ID into the box labeled **Site Id**.
- Select the **Make** button. This will create all four templates containing Station ID, issue and valid times. For example, a 06Z template for KPIT and a 30-h TAF location is:

  KPIT D10520Z D106/D212 =

To modify TAF template:

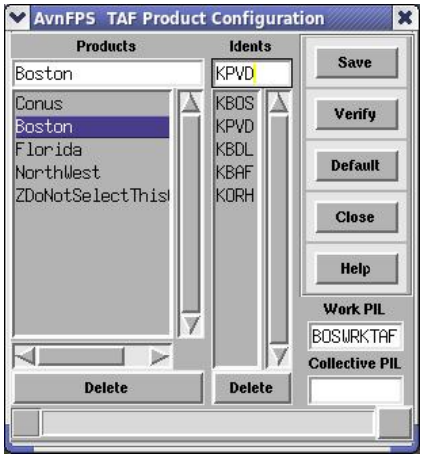
- Enter the Station ID into the box labeled **Site Id**.
- Select issue time from **Issue** menu and press the **Edit** to invoke a text editor. *Do not* press the **Make** button, as this will overwrite your current templates.

Note

With the introduction of 30-h long TAFs, macros `D1` and `D2` make references to the starting and ending day and can be used within the body of the TAF such as `AMD NOT SKED AFT D10300` for example.

4.4. Editing TAF Product Definition

From the **avnsetup** main GUI, select **TAF Products** button. TAF Product Configuration Editor will be shown.



TAF Product Configuration Editor

The existing products will be displayed in the alphabetical order, with the default (if defined) on the top of the list. The **Idents** window displays list of all TAF sites in the product.

To add a product:

- Type the product name in the text window under the **Products** label and press **Enter**. An error message will be displayed that the product cannot be loaded. This is to be expected, as it's being created for the first time.
- Enter Site ID in the window under the **Idents** label. Hit **Enter** after each site.
- Enter PIL for work TAF in the window under the **Work PIL** label. TAFs associated with this product will be saved under this PIL in the text database.
- The **Collective PIL** is *optional*, used by OCONUS sites to send all TAFs listed in the **Idents** text window field under a single text product identifier.
- When finished, press **Save**. A new product definition file will be created.
- The **Verify** button checks whether the AFOS and WMO ID of each TAFs are correct and ready to be transmitted by the **handleOUP.pl**.

To modify an existing product:

- Select the product on the **Products** list.
- To add a new TAF site, type the ID in the window under the **Idents** label and hit **Enter**.
- To delete a site, select it on the **Idents** list and press the **Delete** button.
- Press **Save** to update product definition file.
- The **Verify** button checks whether the AFOS and WMO ID of the new TAF(s) is correct and ready to be transmitted by the **handleOUP.pl**.

To delete a product:

- Select the product on the **Products** list.
- Press **Delete**. A confirmation dialog will be displayed. Press **OK**, this will delete product definition file. TAF site files (info.cfg and any template files) will not be removed.

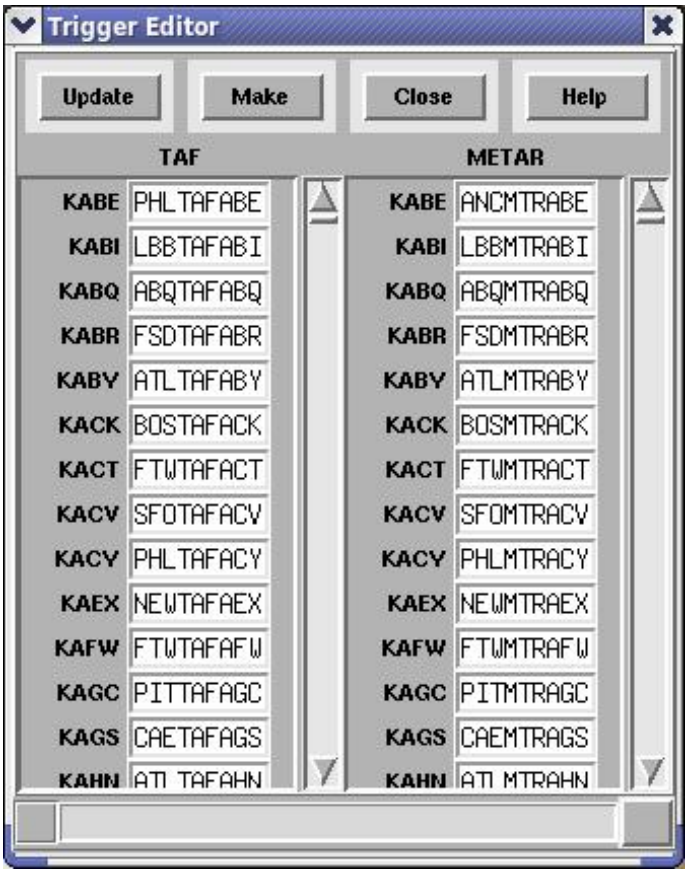
To mark a product as the default:

- Select the product on the **Products** list,
- Press the **Default**. This will designate the selected product as default product when starting the Monitoring GUI.

4.5. Creating Triggers

AvnFPS obtains TAFs, collaborative convective forecast products (CCFPs), and METARs from AWIPS Text Database (`fxatext`). As new versions of these products arrive, the `metarDecoder` can "trigger" another application--usually a script--to run. AvnFPS uses this triggering feature to retrieve TAF, CCFP, and METAR products for monitoring as soon as they arrive in the database. The Trigger Editor manages the configuration of the `textdb` triggers needed to support AvnFPS. The current version creates trigger template, to be processed by AWIPS localization script and simultaneously updates the table `watchwarn` in `fxatext` database holding Postgres trigger scripts. So you do not have to run localization after creating the template file. However the file is still necessary, as the localization process drops the `watchwarn` table and recreates it from templates.

To start the Trigger Editor, select **Triggers** button. The Trigger Editor will be displayed.



Trigger Editor

On startup, the Trigger Editor searches AvnFPS configuration files for all TAF and METAR sites routes that have been configured by your office. (These are files stored in the directories `etc/tafs` and `etc/mtrs`). The two columns: **TAF** and **METAR** list those sites. If there is a template file, `etc/triggerTemplate`, its entries are used to fill the corresponding PIL boxes. If the trigger template does not have an entry for an id, the relevant box will remain blank, with a pink background.

Note

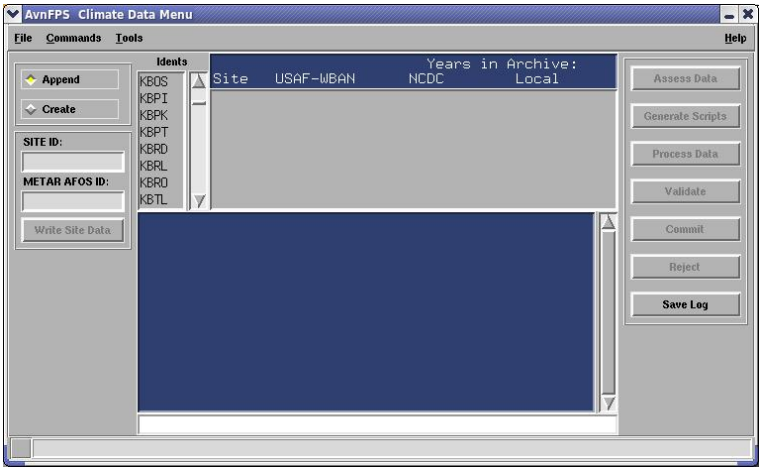
The CCFP product issued from Aviation Weather Center is a national product and not site-specific. This product, consisting of 3 text files, are in the `watchwarn` table, by default.

To create AvnFPS triggers:

- Press the **Update** button. The program will try to determine missing PILs for the TAFs from the file `afos2awips.txt`. METAR PILs are created by from corresponding TAF PILs by replacing `TAF` by `MTR`. The existing values *will not* be overwriten.
- Correct the entries and fill in empty fields.
- Press **Make** when finished. Empty (pink) entries will be skipped, if you have an invalid entry, an error message will be shown. If there are no errors, the editor will save the contents of the editor to a file and update the watchwarn table in the fxatext database.

4.6. Climate Data File Tool

From the **Avnsetup** main GUI, selecting the **Climate Data** button will cause the Climate Data Tool Dialog to appear.



Climate Data Tool Dialog

4.6.1. Introduction

Since AvnFPS 3.0 with the introduction of its climatology tools, an up-to-date archive of METAR observations was needed for each TAF. Given the current count of more than 620 TAFs, it became time consuming to refresh these archives each year at MDL [http://www.mdl.nws.noaa.gov/~avnfps/data/hdf5/index.html]. A better solution was to encapsulate the software that generates the HDF5 climate files into a user-friendly interface and provide that as part of the AvnFPS suite of configuration tools. The Climate Data Tool allows the user to update the AvnFPS climate files as often as they wish, limited only by the availability of latest QC'd observations at the National Climatic Data Center (NCDC).

The Climate Data Tool will allow users to create brand-new HDF5 climate files as well as update the ones you already have. For either task, the steps to perform are nearly the same. Creating brand-new files requires an additional preliminary step. The bulk of the procedure to update or create the HDF5 climate file have been broken down into a series of steps implemented as buttons which are laid out from top to bottom on the right-hand side of the GUI.

The following table briefly describes the purpose and actions of these buttons.

Button	Description/Purpose
Assess Data	Examines existing AvnFPS climate file(s), if present, with NCDC's ISH files to determine what years are needed from NCDC to bring the local file(s) up-to-date.
Generate Scripts	Create and view scripts for downloading NCDC data files for either Linux or Windows computers.
Process Data	Analyze and incorporate NCDC files into HDF5 climate file(s)
Validate	Temporarily move new HDF5 climate file(s) into place for viewing by AvnFPS climate tools for verification
Commit	Move new HDF5 climate file into the AvnFPS database permanently. Perform cleanup actions by removing NCDC files and old HDF5 climate file.
Reject	Restore previous HDF5 climate file.
Save Log	Record session activity to disk.

Note

It is important to recognize that while the steps--briefly outlined in the table above--are few and relatively uncomplicated, this procedure does requires the use of an open Internet facing computer, either a Linux or Windows machine, to access the NCDC site and its archives. The process of transferring scripts and data files to and from AWIPS and the 'outside world' is left as an exercise for the reader.

4.6.2. Adding New Site

If the TAF site in question is not listed in the "Idents" scrolled window, you will need to add it.

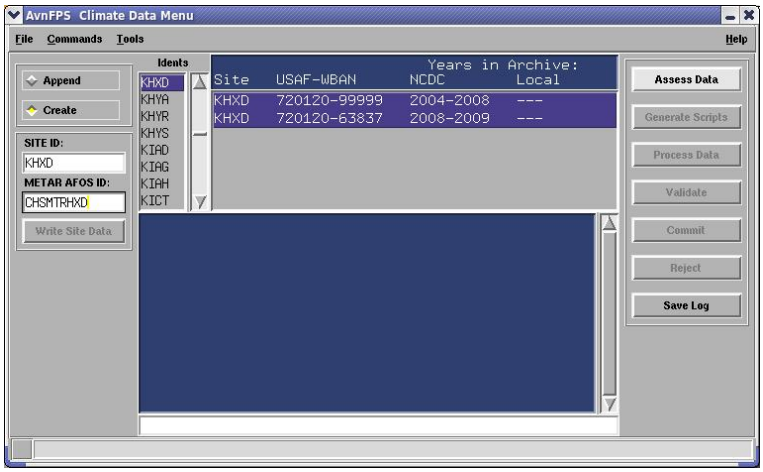
With the **Create** radio button in the Climate Tool Dialog selected, fill in the **Site ID** and **METAR AFOS ID** text fields and then select the **Write Site Data** button to add it to the **Idents** list.

Adding new site

Proceed with the steps in the next section to create the new climate file.

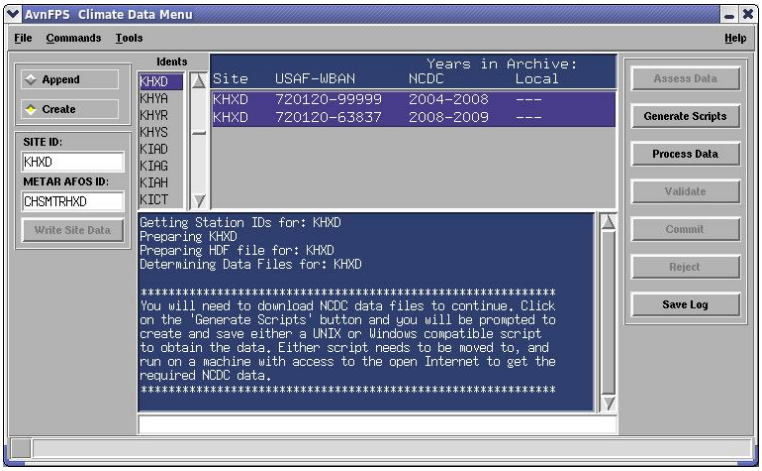
4.6.3. Analyzing Climate Files

With the TAF site selected in the **Idents** window, information about the site(s) selected will appear in the text window. The information displayed includes the site's USAF and WBAN identifiers, and the NCDC data available for these identifer(s). In this case, KHXD is a new location with no local climate file, so NCDC has observations spanning from 2004 to 2008, and the 'Local' column has no information to display, which is denoted by a string of dashes, '---'. For existing climate files in the AvnFPS database, a span of years would be displayed, indicating the file's current contents.



Assessing the HDF5 climate file

After clicking on the **Assess Data** button, the NCDC's ISH files and local HDF5 climate file (in this case, none for KHXD) are examined to determine what is needed to bring the local climate file up-to-date. After the analysis, you will be prompted for the next step.



Climate File Assessment

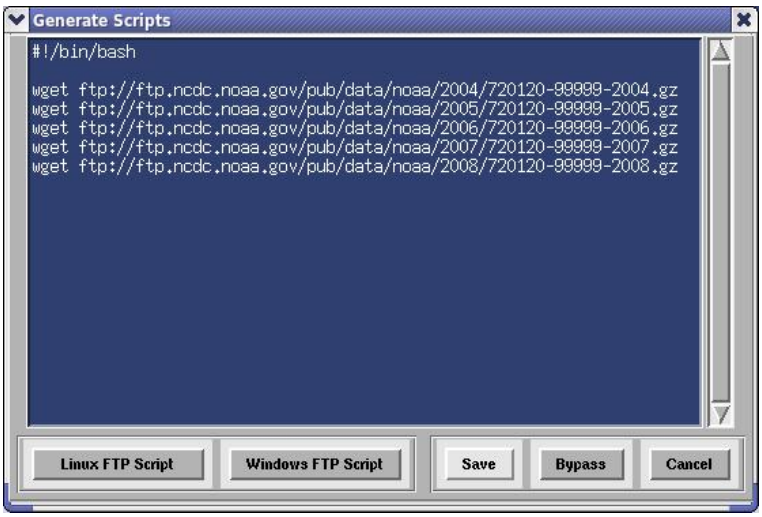
Note

When creating the climate file(s) the software will determine what NCDC files are needed only within the last 30 years. The reasons for this are initial difficulties with memory allocation and length of time to process very long climatic histories. This time restriction may be removed in a later version of this tool and, in any case, does not prevent users from appending later years to existing files.

4.6.4. Downloading NCDC Climate Files

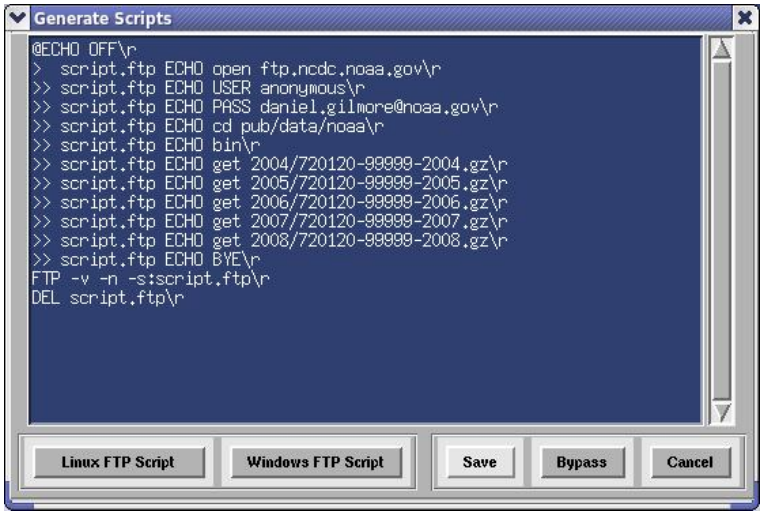
Clicking on the **Generate Script** button will open a dialog allowing you to create either a Windows or Linux FTP script to download NCDC files needed to bring your local climate files up-to-date.

By clicking on the **Linux FTP script**, a script with suitable commands for a Linux platform appears in the text window.



Linux script for downloading NCDC files

By clicking on the **Windows FTP script**, a batch file with suitable commands for a DOS platform appears in the text window.



Windows script for downloading NCDC files

Now move the download file to the appropriate machine with Internet access. Once there, run the script to get the needed data files from NCDC.

What follows is an example of a Linux download session to gather NCDC data for site KHXD.

```
[tyr:...tmp] > ./getNCDCData-KHXD.sh
--17:05:22--  ftp://ftp.ncdc.noaa.gov/pub/data/noaa/2004/720120-99999-2004.gz
           => `720120-99999-2004.gz'
Resolving ftp.ncdc.noaa.gov... 205.167.25.101
Connecting to ftp.ncdc.noaa.gov|205.167.25.101|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /pub/data/noaa/2004 ... done.
==> PASV ... done.    ==> RETR 720120-99999-2004.gz ... done.
Length: 48,108 (47K) (unauthoritative)

100%[=====================================================================================>] 48,108
337.35K/s
17:05:23 (335.49 KB/s) - `720120-99999-2004.gz' saved [48108]

--17:05:23--  ftp://ftp.ncdc.noaa.gov/pub/data/noaa/2005/720120-99999-2005.gz
           => `720120-99999-2005.gz'
Resolving ftp.ncdc.noaa.gov... 205.167.25.101
Connecting to ftp.ncdc.noaa.gov|205.167.25.101|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /pub/data/noaa/2005 ... done.
==> PASV ... done.    ==> RETR 720120-99999-2005.gz ... done.
Length: 147,613 (144K) (unauthoritative)

100%[=====================================================================================>] 147,613
644.33K/s
17:05:23 (642.52 KB/s) - `720120-99999-2005.gz' saved [147613]

--17:05:23--  ftp://ftp.ncdc.noaa.gov/pub/data/noaa/2006/720120-99999-2006.gz
           => `720120-99999-2006.gz'
Resolving ftp.ncdc.noaa.gov... 205.167.25.101
Connecting to ftp.ncdc.noaa.gov|205.167.25.101|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /pub/data/noaa/2006 ... done.
==> PASV ... done.    ==> RETR 720120-99999-2006.gz ... done.
Length: 139,125 (136K) (unauthoritative)

100%[=====================================================================================>] 139,125
610.03K/s
17:05:24 (608.62 KB/s) - `720120-99999-2006.gz' saved [139125]

--17:05:24--  ftp://ftp.ncdc.noaa.gov/pub/data/noaa/2007/720120-99999-2007.gz
           => `720120-99999-2007.gz'
Resolving ftp.ncdc.noaa.gov... 205.167.25.101
Connecting to ftp.ncdc.noaa.gov|205.167.25.101|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /pub/data/noaa/2007 ... done.
==> PASV ... done.    ==> RETR 720120-99999-2007.gz ... done.
Length: 134,053 (131K) (unauthoritative)
```

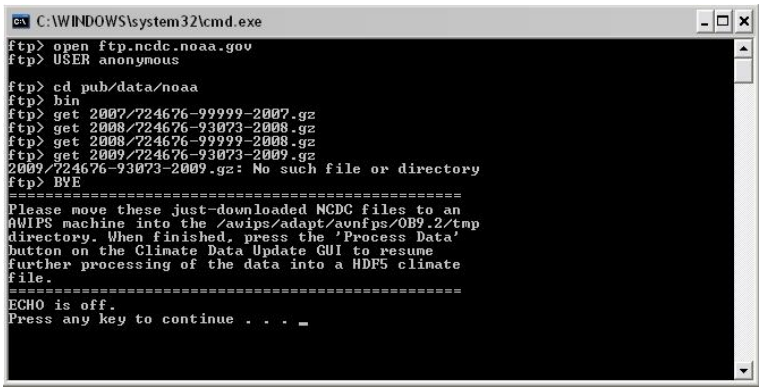
```
100%[===================================================================================>] 134,053
716.04K/s
17:05:25 (713.94 KB/s) - `720120-99999-2007.gz' saved [134053]

--17:05:25--  ftp://ftp.ncdc.noaa.gov/pub/data/noaa/2008/720120-99999-2008.gz
           => `720120-99999-2008.gz'
Resolving ftp.ncdc.noaa.gov... 205.167.25.101
Connecting to ftp.ncdc.noaa.gov|205.167.25.101|:21... connected.
Logging in as anonymous ... Logged in!
==> SYST ... done.    ==> PWD ... done.
==> TYPE I ... done.  ==> CWD /pub/data/noaa/2008 ... done.
==> PASV ... done.    ==> RETR 720120-99999-2008.gz ... done.
Length: 122,064 (119K) (unauthoritative)

100%[===================================================================================>] 122,064
700.17K/s
17:05:26 (699.58 KB/s) - `720120-99999-2008.gz' saved [122064]

==================================================
Please move these just-downloaded NCDC files to an
AWIPS machine into the /awips/adapt/avnfps/OB9.2/tmp
directory. When finished, press the Process Data
button on the Climate Data Update GUI to resume
further processing of the data into a HDF5 climate
file.
==================================================
```

A Windows session after double-clicking on the batch file icon:

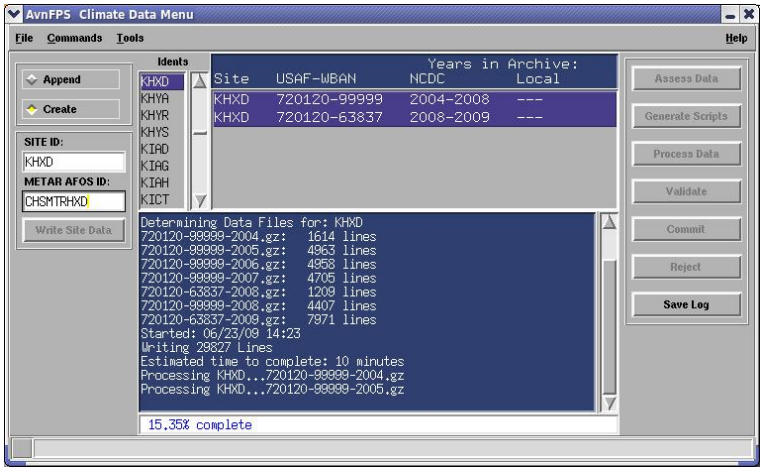


Windows Downloading Session

As the download scripts suggest, copy the downloaded files back into the AWIPS system, to the `/awips/adapt/avnfps/OB9.2/tmp/` directory for continued processing.

4.6.5. Processing NCDC data files

Once the NCDC files have been transferred to the AWIPS machine, clicking on the **Process Data** button will start the actual climate data file update (or creation in this case).



Processing NCDC Data

Once processing is underway, the GUI will provide constant feedback to the user about when the processing started, how many lines are being written, estimated time to complete, how much progress has been made at any moment, and when the processing is complete.

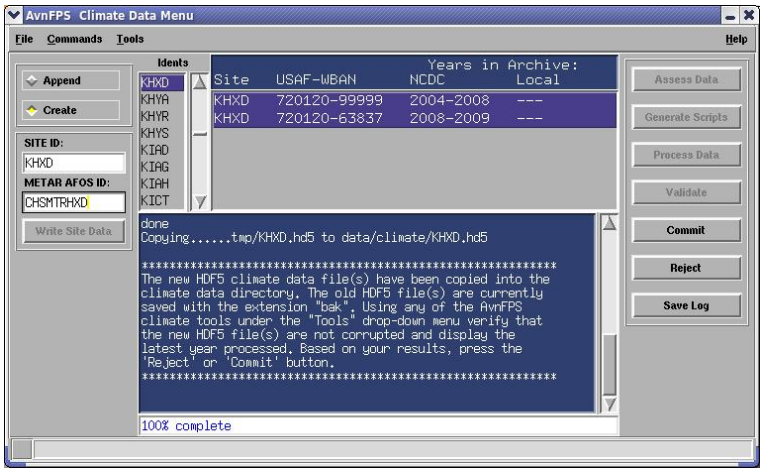
Once processing of the NCDC files is completed, the **Validate** button will become active.

Note

On rare occasions, NCDC archive files can be corrupted. This problem will show up in the text window as decompression errors as processing proceeds. To correct, the user can simply remove the offending file, located in the `/awips/adapt/avnfps/OB9.2/tmp` directory, or try downloading the yearly observation file from the NCDC archives again at a later time.

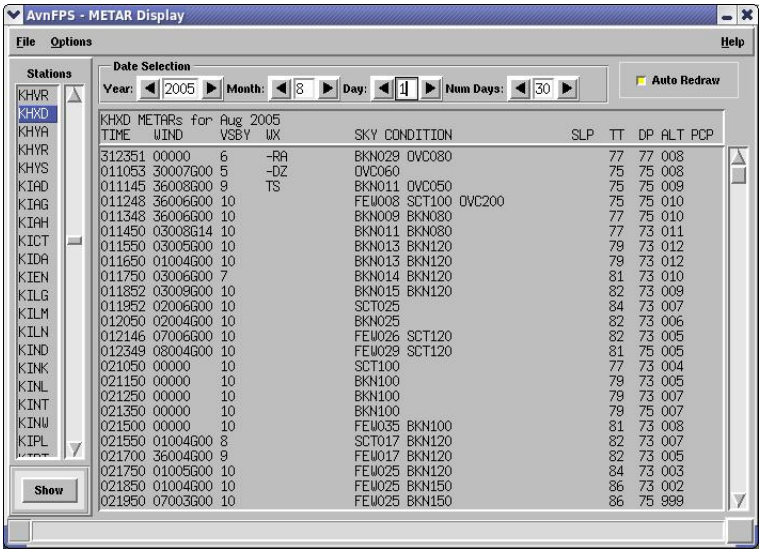
4.6.6. Examining new HDF5 climate file

Once processing is completed, clicking on the **Validate** button will temporarily move the new climate in place so that AvnFPS climate tools can examine the new file. The climate tools can be accessed from the **Tools** pull-down menu.



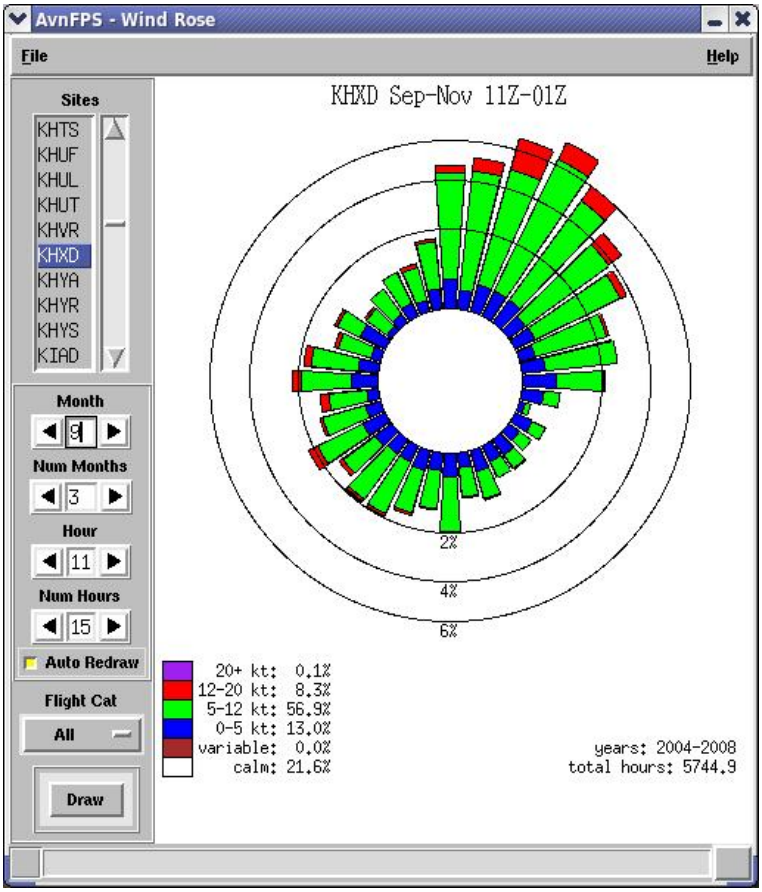
Validating New Climate File

Using the Climate Tool METAR Viewer to examine the new records.



Validating New Climate File with METAR Viewer

Using the Climate Tool Wind Rose to examine the new records.



Validating New Climate File with Wind Rose Tool

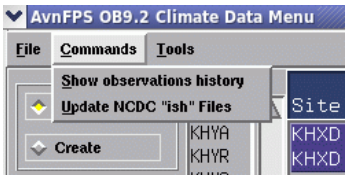
After examining the new climate file using the climate tools and it's found satisfactory, pressing the **Commit** button will keep the new file in place and remove the old one and the downloaded NCDC files.

If the new climate file is not satisfactory (or corrupted), pressing the **Reject** button will restore the previous version of the file back to the AvnFPS database. The downloaded NCDC files remain should you wish to attempt to update the HDF5 climate file again.

The **Save Log** button saves the status text window's content to disk. The log file is stored in the `/awips/adapt/avnfps/OB9.2/tmp` directory.

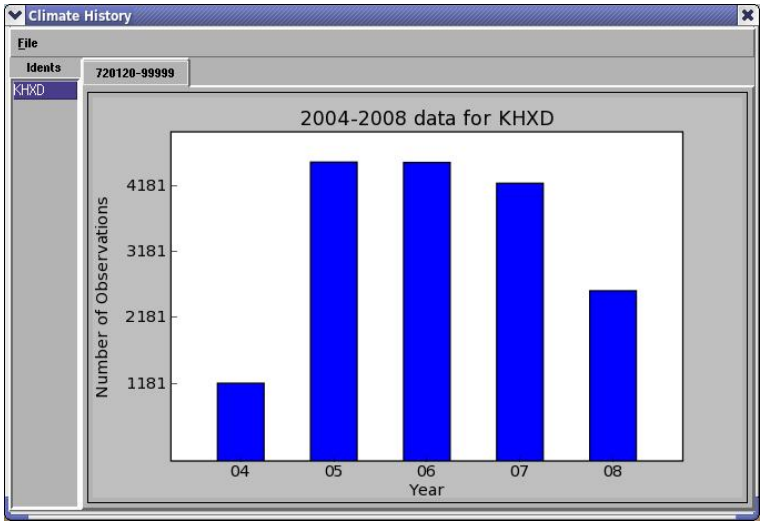
4.6.7. Miscellaneous

In the Climate Data Tool menu bar, selecting **Commands->Data Details** button will open the Data Details dialog.



Climate Data Details

The Data Details dialog presents a bar graph representation indicating the number of observations per year in the NCDC archive files for the selected station.

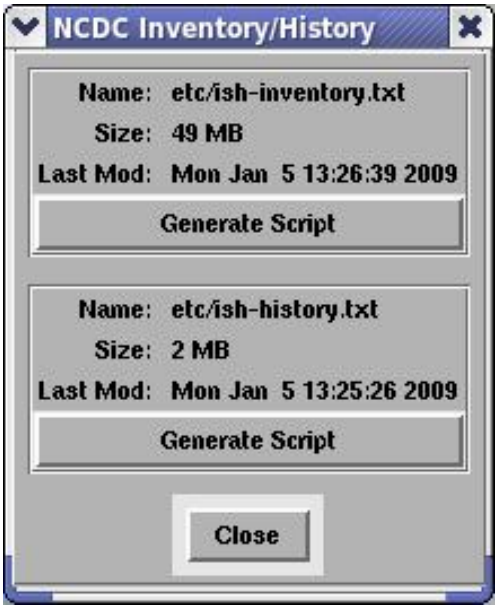


Climate Data Station Details

In the Climate History GUI, there are a few items to note. The Idents list lists all the stations you have selected in the main GUI. Click on one of these stations for histogram(s) representing the number of observations by year available at that station. For each station, one or more tab(s) will appear above the graph. If a station has had multiple USAF-WBAN IDs, then a tab with a separate graph for each USAF-WBAN ID combination will appear.

In the image above, there is just one tab/pane. The visible graph tells us that METAR data for KHXD started in 2004 and continues to 2008.

Selecting **Commands->Update Helper Files** opens the Helper File Info dialog.



Climate Data Helper File Info Dialog

These two data files are crucial to the proper operation of the Climate Data Tool:

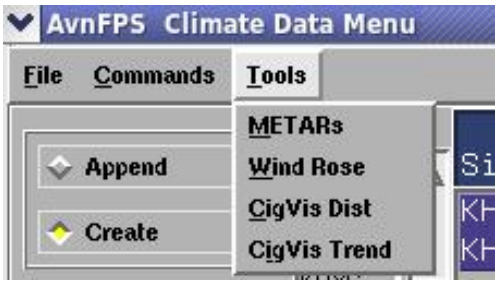
- **ish-history** contains detailed historical information about each station
- **ish-inventory** contains details about the inventory of archived observations available for all stations

These files are regularly updated and made available by NCDC. They should be refreshed at least yearly so as to include the current year when assessing the climate files for updates. The **Generate Script** buttons invoke the dialog which allows the user to generate download script for these files, similar to the data archives.

Note

These 'ish-' files are kept in the `/awips/adapt/avnfps/OB9.2/etc` directory.

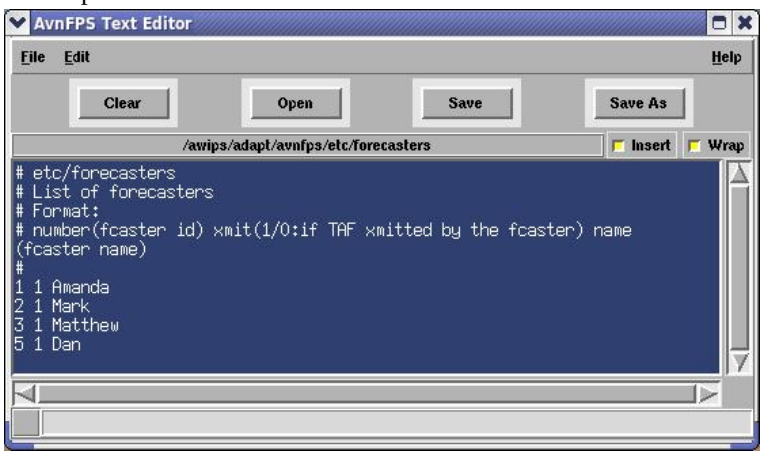
In the Climate Data Tool menu bar, selecting the **Tools** pull-down menu reveals the list of the AvnFPS climate tools. This is provided as a convenience for validating new and updated climate files.



Climate Data Helper Tools Pulldown Menu

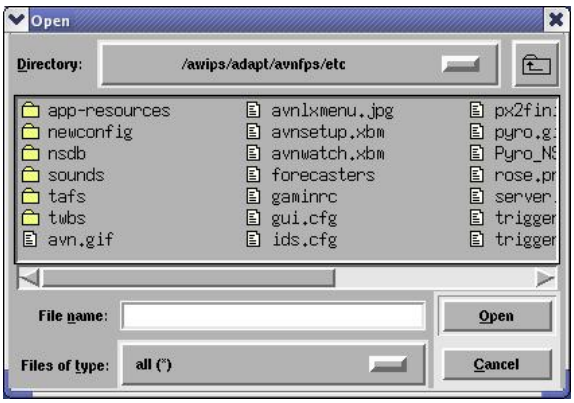
4.7. Using Text Editor

To edit the configuration files that do not have a GUI interface: `forecasters`, `logging.cfg`, `server.cfg`, `gui.cfg`, `ids.cfg`, `flt_cat.cfg`, `grp_taf.cfg`, use the AvnFPS text editor. It is invoked by pressing **Text Edit** button. Here's a snapshot of the editor with the contents of the `etc/forecasters` file.



Setup Editor Dialog

Press **Open**, you will get a file selection dialog listing content of directory `etc`. In this dialog, select a file to edit, then press **Open**.



File Selection Dialog

Press **Save** button when finished. If you want to create a new file, simply type its contents in the text window, and then use **Save As** button to save to disk.

You can also use the text editor to edit X resource configuration file. However, we suggest that you only modify the system-wide file which is used as a initial selection when a forecaster modifies his own resources through the graphical editor available in **AvnWatch**. Proceed as above, and in the file selection dialog, select the `app-resources` folder, then the file `X`. You may edit files for individual forecasters (those with names X.*N*), however any comments you put there will be stripped by the graphical editor every time the forecaster modifies the resources from **AvnWatch**.

Tip

Font and color names may be cryptic. You may want to use the graphical editor in **AvnWatch** first to display color codes and available fonts.

5. Configuring IFPS to send gridded data to AvnFPS

With arrival of OB7.1 and DS decommissioning, it was decided not to port the IFPS applications, **ifpServerWatcher** and **avn_unldr** to use Postgres database resources. These two applications have been retired since OB7.1. Their job was to monitor GFESuite's **ifpServer** for changes in the gridded database, and when notified of such changes, extract gridded data and place it in a format accessable to AvnFPS data ingest server. In their stead, a new **GFE** text formatter was created to do the job of accessing IFPS gridded database and writing the information to the `/data/adapt/avnfps/grids` directory.

The new **GFE** text formatter, **IFPS2AvnFPS.py** is run at least 10 times an hour via cron on the `px2f` machine.

With IFPS software, the process of getting gridded data into AvnFPS was event-driven: triggered when a forecaster saved aviation-related parms in **GFE**. That has been replaced by scheduled-driven process using "fxa" cron. Ultimately, this new technique should be event-driven as well. This enhancement is planned for **ifpServer**.

6. Logging

AvnFPS has an extensive logging system, similar to collective logging in AWIPS. The amount of information written to log files and its format are configurable.

Each program has its own log file, one per day of the week. File naming is according to the scheme *`progname_DayOfWeek`*, so **avndis** will produce files `avndis_Mon` through `avndis_Sun`. Files are broken (old closed, new opened) at the first write attempt of each new day. This is different from AWIPS where a cron job breaks the log after 00Z.

The file `etc/logging.cfg` controls the location, format and log level for the log files. The format and log level is the same for all programs.

Note

The log directory is local to each server/workstation. The directory `/awips/adapt/avnfps/OB9.2/logs` points to `/data/logs/adapt/avnfps`, which resides on a local partition. This is consistent with the AWIPS standard:

```
lx2-wfo:46: ll /awips/adapt/avnfps/OB9.2/logs
lrwxrwxrwx    1 fxa      fxalpha        23 Oct  8 16:41 /awips/adapt/avnfps/OB9.2/logs ->
/data/logs/adapt/avnfps
```

Here is an example listing of a log file for **avndis**:

```
Log rollover at 00:00:01
2009-06-08 00:00:01,280 DEBUG [10517:126217136] TriggerThread: Processing PHLMTRMDT
2009-06-08 00:00:01,647 INFO  [10517:126217136] TextThreadP: Processed KMDT from SAUS70 KMDT 080000
2009-06-08 00:00:01,647 DEBUG [10517:126217136] TriggerThread: Done PHLMTRMDT
2009-06-08 00:00:01,649 DEBUG [10517:126217136] TriggerThread: Processing PITMTRDUJ
2009-06-08 00:00:01,656 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'KMDT'}
2009-06-08 00:00:01,754 INFO  [10517:126217136] TextThreadP: Processed KDUJ from SAUS70 KDUJ 080000
2009-06-08 00:00:01,755 DEBUG [10517:126217136] TriggerThread: Done PITMTRDUJ
2009-06-08 00:00:01,756 DEBUG [10517:126217136] TriggerThread: Processing BUFMTRART
2009-06-08 00:00:01,981 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'KDUJ'}
2009-06-08 00:00:01,989 INFO  [10517:126217136] TextThreadP: Processed KART from SAUS70 KART 080000
2009-06-08 00:00:01,990 DEBUG [10517:126217136] TriggerThread: Done BUFMTRART
2009-06-08 00:00:01,993 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'KART'}
2009-06-08 00:00:05,289 DEBUG [10517:126217136] TriggerThread: Processing HFOMTRHNY
2009-06-08 00:00:05,576 INFO  [10517:101030832] LtgThread: Processing file
/data/fxa/point/binLightning/netcdf/20090607_2300
2009-06-08 00:00:05,629 INFO  [10517:126217136] TextThreadP: Processed PHNY from SAUS80 PHNY 080000
2009-06-08 00:00:05,631 DEBUG [10517:126217136] TriggerThread: Done HFOMTRHNY
2009-06-08 00:00:05,633 DEBUG [10517:126217136] TriggerThread: Processing BUFMTRROC
2009-06-08 00:00:06,020 DEBUG [10517:27261872] DataIngestServ: Msg: {'src': 'mtrs', 'ident': 'PHNY'}
```

Each line contains the following fields:

- Date and time of the event.
- Severity level (in caps).
- Process and thread number (in square brackets).
- Python module name which generated the message.
- Actual message content.

Some typical errors:

```
2009-06-08 02:00:48,843 INFO  [10517:84941744] MosData: Retrieved NAM data for KBUF
2009-06-08 02:00:48,851 ERROR [10517:84941744] MosData: KBVI not in
/data/fxa/point/mos/ETA/netcdf/20090608_0000 ❶

2009-06-08 02:00:48,882 INFO  [10517:84941744] MosData: Retrieved NAM data for KCHA

2009-06-08 05:32:52,556 DEBUG [10517:126217136] TriggerThread: Done MEMTAFTRI
2009-06-08 05:32:52,558 WARNING [10517:74451888] util: Daemon ** Exception during processing of request from
TCPConnection with ('xxx.xxx.xxx.xxx', 45051)
connected=1  type thread.error ❷

--- traceback of this exception follows:
 ---------------------------------------------------
 <thread.error> RAISED : can't start new thread

 Extended Stacktrace follows (most recent call last)
---------------------------------------------------
File "/awips/adapt/avnfps/Python-2.4.1/lib/python2.4/site-packages/Pyro/protocol.py", line (922), in
Daemon::connectionHandler
Source code:
    self.handleInvocation(conn)
File "/awips/adapt/avnfps/Python-2.4.1/lib/python2.4/site-packages/Pyro/core.py", line (695), in
Daemon::handleInvocation
Source code:
```

❶ This message indicates problem with TAF Site Info configuration file: data for KBVI is not distibuted for NAM MOS product. This does not cause any harm, but one can either select another site which is close enough, or leave the NAM MOS site id empty.

❷ Some messages provide detailed traceback to point out to the exact location of the code where the error occured. In this case, the error comes from Pyro code and it is relatively serious. The data ingest server hit the threshold of allowable memory consumption. **avndis** has this threshold set programatically to avoid clogging the whole operating system. In a situation like this one should restart **avndis** to avoid data loss. If the problem reoccurs, further investigation is required. (This example is contrived: the threshold was set too low during development).

7. Troubleshooting

Techniques for troubleshooting various aspects of AvnFPS are addressed in other sections of this manual. The purpose of this section is to gather some of the more likely fault scenarios and troubleshooting techniques into a single location.

7.1. Overall Server Health

A number of server processes keep AvnFPS running. A series of status lights on the TAF Monitor GUI reports the status of most of these processes. Keep alive messages are sent among the servers every 30 seconds to set the colors of these status lights. Section 1. of the System Administration Manual, "System Overview," contains a detailed description of each server. Server hosts may vary between AWIPS releases to support load-balancing efforts. For AWIPS OB9.2, the servers reside on a single host, `px2f`.

7.1.1. Diagnostic Tools

- If all the server status lights are red, check for
  - Large (>60s) clock skew between client and server hosts, nominally between px2f machine and AWIPS workstations.
  - Failed name/event server (avnserver)
  - A network failure
- The command **`ps -ejHf | grep avnpython`** lists server processes and any component threads. In this case, indentation in the rightmost column indicates hierarchy. A sample listing follows:

```
px2-wfo:user: ps -ejHf | grep avnpython
fxa 18202     1 18201 18201 0 Apr28 ? 00:00:00  avnpython /awips/adapt/avnfps/OB9.2/py/avninit.py px2f
fxa 18203 18202 18201 18201 0 Apr28 ? 00:06:46   avnpython /awips/adapt/avnfps/OB9.2/py/avnserver.py -d -n
px2f
fxa 18224 18202 18201 18201 0 Apr28 ? 00:02:10   avnpython /awips/adapt/avnfps/OB9.2/py/avndrs.py -d -n
px2f
fxa 18239 18202 18201 18201 0 Apr28 ? 00:17:02   avnpython /awips/adapt/avnfps/OB9.2/py/avndis.py -d -n
px2f
fxa 18250 18202 18201 18201 0 Apr28 ? 00:00:55   avnpython /awips/adapt/avnfps/OB9.2/py/avnxs.py -d -n px2f
```

  Notes:

  - The command given above will shows the actual processes. To see the threads spawned by those processes use **`-efL`** flags to the ps command: **`ps -efL | grep avnpython`**.
  - Some comments follow:

PID	Comments	"Kill"-able?
18202	avninit daemon	Yes
18203	avnserver process	Yes
18224	Data Request Server (avndrs) process	Yes
18239	Data Ingest Server (avndis) process	Yes
18250	Transmit Server (avnxs) process	Yes

  - Threads of a process will not accept signals from a kill command. You must identify the process that owns the thread and kill that top-level process. Once a parent process terminates, all of its threads will terminate as well.
- The command **`netstat -a | grep 9090`** can be used to diagnose connections among the various servers. The name/event server uses port 9090 to disseminate information about the various servers that are up and running. Here's an example:

  Checking netstat on px2 …

```
px2-wfo:user: netstat -a | grep 9090
tcp        0      0 px2f-wfo:9090             px2f-wfo:55812             ESTABLISHED
tcp        0      0 px2f-wfo:9090             px2f-wfo:55819             ESTABLISHED
tcp        0      0 px2f-wfo:55819            px2f-wfo:9090              ESTABLISHED
tcp        0      0 px2f-wfo:55812            px2f-wfo:9090              ESTABLISHED
tcp        0      0 px2f-wfo:9090             lx3-wfo:40638              ESTABLISHED
```

Notes:

- The columns are Protocol, Receive Queue, Send Queue, Local Address, Foreign Address, and State.
- Six connections to and from the name/event server can be seen in the `px2` listing.
  - The first four established connections show connections between the servers on `px2` and the name/event server.
  - There is an additional connection to a process on `lx3`. Most likely, this is an instance of the **AvnWatch** GUI.

7.1.2. Stopping/Starting Servers

avninit

Starting the servers for AvnFPS is a little complicated because of the inter-relationships among them. The utility **avninit** was developed to handle these complexities. **avninit** is a persistent process that runs on `px2` or, in case of failover, `px1` and attempts to restart servers as needed. If you fix a file or network problem that is preventing a server from starting, **avninit** will attempt to restart the downed server. However, **avninit** will only attempt 10 restarts of a failed server in a one-hour span of time.

Environment

Important

All AvnFPS servers must run at user fxa; do not try to start them as root. The standard server startup scripts will check userid before launching the servers.

Avnkill

The utility **avnkill** can be used to stop all AvnFPS servers that are running on a host. This includes **avninit**. **avnkill** will be persistent, sending interrupt signals first, then kill signals. It will also try to clean up ill-behaved child processes that refuse to die when the parent dies. Once all servers have stopped running, restart **avninit**, using the **remoteServers.sh start**

Bouncing Servers

To "bounce" a Data Ingest, Data Request, or Transmisison Server, identify the top-level process associated with the application and use the kill command to stop it. Within a few seconds, **avninit** should launch a new instance.

7.2. Log Files

See Section 6: “Logging” (page 54) of the System Administration Manual, for complete information on logs. The following information is distilled from that section.

Log files for AvnFPSOB9.2 server processes are stored in the directory tree `/data/logs/adapt/avnfps`, local to the host computer. The names of the logs files are formed by the name of the application and the current day of the week (e. g., `avnmenu_Thu` and `avndis_Fri`). All applications use collective logging which means that log file entries from different instances of an application can be found, interleaved, in a single log file. The following screen capture shows sample directory listings:

GUI logs on a workstation
…

```
lx2-wfo:user: pwd
/data/logs/adapt/avnfps
lx2-wfo:user: ls
avnclimate_Fri  avnmenu_Mon  avnmenu_Thu  avnqcstats_Tue  avnsetup_Thu  avnwatch_Mon  avnwatch_Thu
avnclimate_Wed  avnmenu_Sat  avnmenu_Tue  avnsetup_Mon    avnsetup_Tue  avnwatch_Sat  avnwatch_Tue
avnmenu_Fri     avnmenu_Sun  avnmenu_Wed  avnsetup_Sun    avnwatch_Fri  avnwatch_Sun  avnwatch_Wed
```

Server logs on px1/px2 …

```
px2-wfo:user: pwd
/data/logs/adapt/avnfps
px2-wfo:user: ls
avndrs_Fri  avndrs_Sun  avndrs_Wed   avninit_Sun  avninit_Wed    avnserver_Sat  avnserver_Tue
avndrs_Mon  avndrs_Thu  avninit_Fri  avninit_Thu  avnserver_Fri  avnserver_Sun  avnserver_Wed
avndrs_Sat  avndrs_Tue  avninit_Sat  avninit_Tue  avnserver_Mon  avnserver_Thu
```

It is possible for certain unexpected errors to go undetected in log files. One way to test against this possibility is to launch a GUI process from the command line and watch the standard error and standard output streams. Here is a command that will launch the AvnFPS startup menu from the command line:

```
lx2-wfo:ncfuser:19: /awips/adapt/avnfps/bin/avnstart.sh avnmenu
```

7.3. Data Ingest

Data Ingest Servers (instances of **avndis**) monitor events in AWIPS and decode various data as they become available. For some data types, the term decode means little more than copying a file. Once avndis has put the data into the AvnFPS directory tree, Data Request Servers (instances of **avndrs**) serve the data to the various GUIs.

7.3.1. Data Ingest Troubleshooting

- Log files for avndrs show data being delivered to GUIs as well as data access problems. See if GUIs are connecting to avndrs successfully and receiving data.
- Log files for avndis show data arriving and being decoded. Errors are rather easy to spot in here.

7.3.2. Specific hints for specific data types

Text (TAFs, METARs)	Database triggers deliver products to `/awips/adapt/avnfps/data/text`.
Lightning Observations	Directory `/data/fxa/point/binLightning/netcdf` is monitored for new and updated files.
Lightning Probability	Directory `/data/fxa/img/SBN/netCDF/LATLON/3hr/LTG` is monitored for new and updated files.
Low-Level Wind Shear	The following directories can be monitored for new files: • `/data/fxa/point/acarsProfiles/netcdf` • `/data/fxa/point/profiler/netcdf` • `/data/fxa/LDAD/profiler/netCDF`, • `/data/fxa/radar/`*`CCCC`*`/VWP`, where *`CCCC`* = radar-ID
IFPS Grids	An entry in `/etc/cron.d/px2apps` issues the job that exports the data from GFESuite ifpServer to `/awips/adapt/avnfps/data/grids` where the files are monitored by gamin.

7.4. Product Transmission

Product transmission is covered in depth in Section 1.4: “Transmission Server” (page 7) of the System Administration Manual. Here are the recommended steps to follow:

- Check log file on px2f for **avnxs** server normally located in the `/data/logs/adapt/avnfps` directory. If a forecast prepared by AvnFPS was written to the pending queue `/awips/adapt/avnfps/OB9.2/xmit/pending` and the transmission server attempted to acces the file, there should be a corresponding entry, starting with the word `SUCCESS` or `FAIL`*`NNN`*, where *`NNN`* is the code returned by the system call to **handleOUP.pl**.
- If **avnxs** indicates **handleOUP.pl** failure, proceed to investigate its log file, `/data/logs/fxa/`*`yyyymmdd`*`/handleOUP.log`, on the px2f machine,.

Chapter 2. User's Guide

This chapter is intended for forecasters using AvnFPS to monitor and prepare aviation forecasts.

1. Startup Menu

To start the AvnFPS application, you can start with the AvnFPS startup menu. The menu can be invoked from the command line by typing

```
lx2-wfo: /awips/adapt/avnfps/bin/avnstart.sh avnmenu
```

However, most users will start it from the AWIPS desktop popup menu.

AWIPS application launcher

When **AvnMenu** is started, you'll see a window as in the example here:

AvnFPS startup menu

The top list displays the names (or initials) of forecasters at your office. This name list permits individual customizations for things such as colors and fonts. The list at the bottom of the menu displays data server hosts than

can be used to supply data to AvnFPS. For normal operations, the option named **local** should be selected.

To launch the TAF Monitor:

- Select your name/initials from the top list.
- Ensure **local** is highlighted on the lower list.
- Select **TAFs** button

To launch the Climatology Menu GUI:

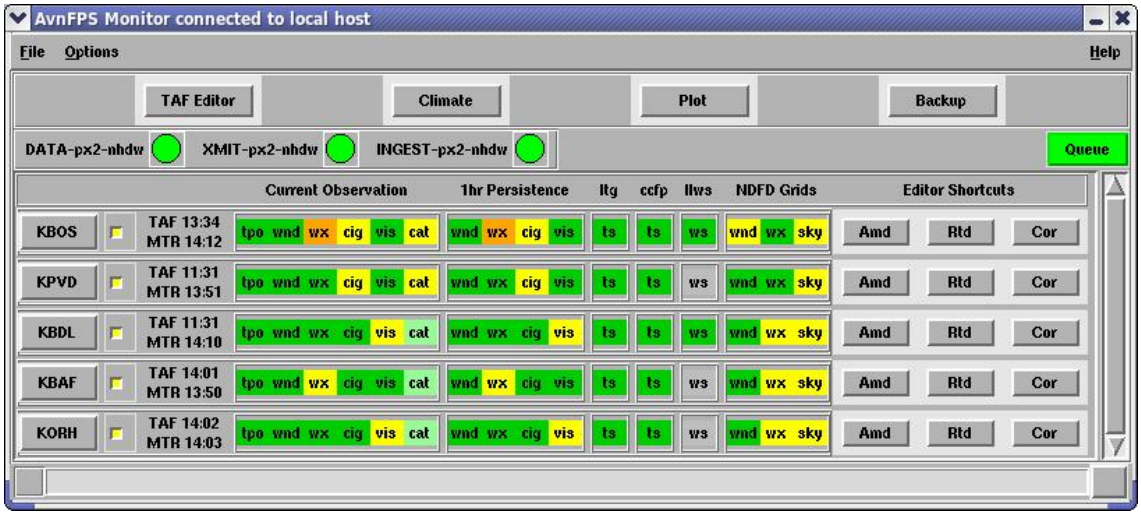
- Ensure **local** is highlighted on the lower list.
- Select **Climate** button

2. TAF Monitoring

The AvnFPS TAF Monitor (aka **AvnWatch**) serves several functions.

- It displays information about the status of background services that are critical for correct functioning of AvnFPS.
- It displays results of monitoring TAF forecasts against observations and various guidance sources
- It provides access to a 'smart' TAF editor.
- It allows you to customize the look of AvnFPS Monitoring GUI.

The image below shows the AvnFPS TAF Monitor.



AvnFPS Monitor

The TAF Monitoring GUI, more popularly known as "AvnWatch", includes one row of indicators and labels for each TAF site of interest to your office. AvnFPS reviews the latest guidance, forecasts, and observations and reports the results in the color-coded portions of the window. Above these TAFs is a row of server status indicators. AvnFPS uses background processes, i.e. servers, running on one or more hosts to accomplish its tasks. The editor and product selection buttons (on the right-hand side) are arrayed on the Monitor as well. These buttons allow to quickly begin your routine and/or unscheduled product generation tasks. The menu bar allows you to access the user configuration tool for AvnWatch. Most of the user configuration items relate to the "look and feel" of AvnWatch on your workstation. Site-level configuration (adding or removing a TAF site, change alert criteria, etc.) is accomplished with a different set of applications.

The message bar at the very bottom shows messages that do not require forecaster intervention (if intervention is required, a popup dialog is displayed). When a text message is displayed for a short period of time, the label in the left lower corner flashes with color indicating severity of the message, which usually is one of the following (the colors are configurable): green: info, orange: warning, red: error, . The time the message is shown depends on the severity and varies between 5 seconds for informational message to 20 seconds for system errors. You can view all messages that have been displayed by pressing the unlabeled button in the right lower corner.

2.1. TAF Monitoring GUI, aka "AvnWatch"

The TAF Monitoring GUI contains rows of buttons and labels for all the TAF sites that have been configured for your office. Each row provides you with some information about the current TAF and how it compares to the latest observations or guidance. A few of the row items act as controls for the monitoring process.

GUI Elements

Site ID Button

These buttons are on the left-hand side of the TAF Monitoring GUI. Clicking on these buttons invokes TAF Viewer. Any data that changes colors in TAF's status (see below) causes the Site ID button to blink for 5 minutes (unless disabled). TAFs containing errors are indicated by Site ID button's background color changing to orange. Missing or un-decodable TAFs or METARs will make the background color of the Site ID button red. To stop the blinking, or to revert to the default background color, right click on the Site ID button.

Active Toggle Button

The small square between the **Site ID** Button and the TAF/METAR Time Labels allows you to enable/disable monitoring of a TAF. When the box is selected (yellow), the current TAF for that site will be monitored.

TAF/MTR Time Labels.

This label displays issuance times of the most recent TAF and METAR. The background color is used to notify the forecaster if one or both reports are late. When the last received report is older than the threshold value, the background changes to orange. The threshold value is 65 minutes for METARs and 6 hours 40 minutes for TAFs.

Observation/Guidance Status

For each data source being compared to the TAF, one or more weather elements are monitored, using a set of rules configured by your office. Each element has a dedicated label. These are color coded, green if the element passed all the rules, gray if data were missing, and one of the following colors: pale green, yellow, orange, red and purple if one or more rules were violated. The color of the label suggests the severity of the violated rule with pale green being least severe and purple being most severe. The following data sources are available for monitoring:

Current Observation

The weather elements under the Current Observation heading represent a comparison of the current METAR observation with the current TAF. The five items are interpreted as follows:

tpo	TEMPO	An alert is displayed for this item if the current TAF contains a TEMPO group and the conditions described in that TEMPO group have not been observed.
wnd	Wind	Alerts on discrepancies in wind speed, direction and, if configured, runway cross- and head-wind components.
vis	Visibility	Alerts on discrepancies in visibility.
wx	Weather	Alerts on discrepancies in precipitation and/or obstruction to vision.
cig	Ceiling	Alerts on discrepancies in ceiling.
cat	Flight Category	Alerts on discrepancies in flight categories.

NUM-hr Persistence

The weather elements under this heading compare the current TAF for *NUM* hours from now with the current METAR observation. The same rules are applied to the weather elements under this section of AvnWatch and the **Current Observation** heading. In effect, these columns assess the current TAF against the assumption that conditions in the latest surface observation will persist for *NUM* hours. The value of *NUM* is selectable: move the mouse over the 'persistence' label and right-click, a popup menu appears which allows you to select the number of hours. Weather elements and alerts are the same as for Current Observation section, with the exception of the

TEMPO check.

ltg
The weather element under this heading represents a comparison of the current TAF with the latest data from the lightning detection network.

ts	Number of strikes	This item will display alerts when lightning is observed near the TAF site and TS is not mentioned in the TAF.

rltg
The weather element under this heading represents a comparison of the current TAF with the latest 4-hr thunderstorm probability forecasts from the Local Aviation MOS Program (LAMP) system. LAMP forecasts are generated hourly by the National Centers for Environmental Prediction Center. See [[Charba, et al.] [131]]. LAMP thunderstorm probability guidance is updated every hour.

ts	Lightning probability	This item will display alerts when lightning probabilities are high near the TAF site and TS is not mentioned in the TAF or vice versa.

ccfp
The Collaborative Convective Forecast Product is a graphical representation of expected thunderstorm occurance at 2-, 4-, and 6-hours after issuance time. The forecast is disseminated as a text product and stored in AWIPS fxatext database. AvnFPS decodes it, finds whether the monitored TAF sites fall within a convective area and alerts whenever convection parameters: coverage and confidence exceed customized thresholds and there is no thunderstorm in the relevant period(s) of the TAF. See [[CCFP] [131]] for description of the CCFP message.

ts	Convection potential	This item will display alerts when convection is forecast, but TS is not mentioned in the TAF. The check is done for all 3 forecast times.

NDFD Grids
The gridded forecasts of weather, wind and total sky cover prepared for the National Forecast Digital Database (See [[Glahn, et. al.] [131]]) are examined by AvnFPS for the consistency between the TAF and the nearest gridpoint.

Note

Remember that the data displayed in the NDFD Guidance Window are generally representative of a larger area (2.5x2.5km^2 or 5x5km^2, depending on WFO grid resolution resolution) than the TAF.

The guidance prepared for NDFD is split into 3 groups:

wnd	Wind	Alerts for discrepancies in wind speed and/or wind direction.
wx	Weather	Alerts for discrepancies in weather.
sky	Sky cover	Alerts for discrepancies in sky cover.

llws
The weather element under this heading represents a comparison of the current TAF with the latest wind profile data captured from suitably equipped aircraft, or NWS's WSR-88D's or FAA Terminal Doppler Weather Radar's radar products as well as the NOAA Profiler Network.

ws	Wind shear value	This item will display alerts when wind profile data indicate wind shear is present near the TAF site and wind shear is not mentioned in the TAF

Note

Colors displayed in the weather element labels, data sources and menus are configurable by office and by user, see System Administration Manual, the section called “`etc/gui.cfg`” [13].

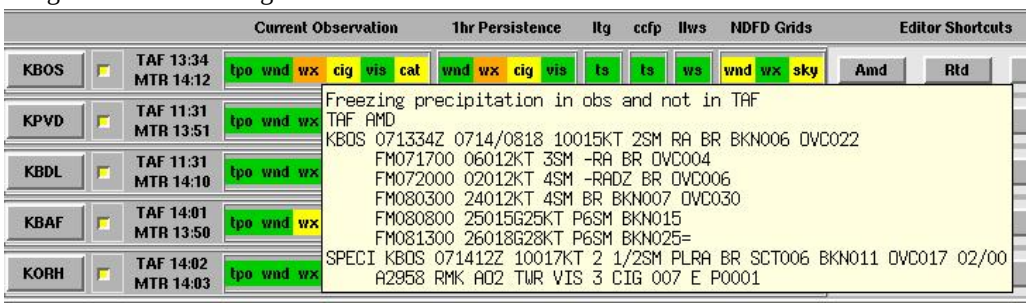
Editor Shortcuts

If you have configured AvnFPS to show the Editor Shortcuts (see Section 2.5, “Resource Editor” [75], the relevant resource is `*amdbuttons`), you will see three buttons to the right of the monitoring indicators. These buttons allow you to quickly launch the TAF Editor to prepare amended, corrected, or routinely delayed forecasts.

Common Actions

To view monitoring details:

Hover over the weather element indicator: a popup window will show displaying current TAF, observation/guidance and messages for all violated rules.



To view observations and guidance for a TAF site:

Select the ID button.

To stop the Site ID buttons from blinking:

Right-click the Site ID button.

To change the number of hours used in persistence monitoring:

Right-click on the persistence label, then select the number of hours from the pop-up menu that appears.

2.2. Server Status Indicators

The Server Status Indicators provide you with important information about AvnFPS's ability to ingest data and transmit products. Similar to the open source Linux-HA [http://www.linux-ha.org] software, AvnFPS servers are constantly monitored by the **avninit** process, and if they fail, **avninit** will immediately re-start another instance. The servers, in turn, broadcast ALIVE messages to listening clients periodically. If such messages aren't received by clients within a short interval, the affected server indicator(s) will turn red. Such cases are indicative of a serious problem and your Aviation Focal Point or ITO should investigate the cause. Remember that the servers and its log files reside on the `px2f` machine. If data or ingest servers have failed, the monitoring functions of AvnFPS will be unreliable or fail completely. If the transmission server has failed, AvnFPS will not be able to issue products. Common causes for failures include:

- A misconfiguration of system files,
- Computer or network maintenance work,
- Failures in system software or hardware,
- Large system clock differences between `px2f` and other workstations.

If your aviation focal point and/or ITO is unavailable for assistance, the NCF should be contacted.

Tip

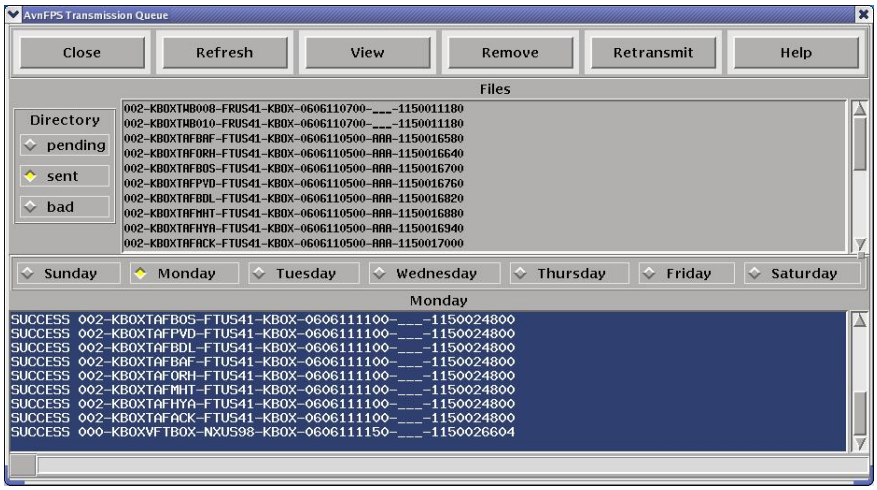
If all indicators are red, then its most likely there is a failure in the local network, or a significant clock skew between server machine and forecaster's workstation, and not a problem with AvnFPS *per se*. Once the network or system clock problem is resolved, the aviation focal point or ITO should restart the AvnFPS servers.

2.3. Transmission Queue

The **Queue** button to the right of the Server Status Indicators brings up another GUI that allows you to view and modify the forecast transmission queue.

Product Transmission Queue Button

The background color of the **Queue** button reflects the status of the most recent transmission attempt: green for success, red for failure. Selecting the **Queue** button launches the AvnFPS Transmission Queue Dialog. A sample Transmission Queue Dialog is shown below:



Transmission Queue Dialog

The top window of the Transmission Queue Dialog, in this image because the 'sent' option has been selected, displays a list of files successfully sent out over the WAN. A set of radio buttons to the left side of window changes the display: you can see products waiting in the queue, 'pending' (often empty), products successfully disseminated, 'sent' (as in this case), or the 'bad' files - either products that did not go out, or a rogue file that should not have been put in the queue. The bottom window displays the results of all transmission attempts for today to date. The row of buttons labelled with the days of the weeks allow you to control the day for which the transmission log is displayed. AvnFPS maintains one week of logs.

The list below describes the actions of the buttons on the top row of the AvnFPS Transmission Queue Dialog:

Close
: Closes the viewer.

Refresh
: Updates the information in the windows. This does not happen automatically.

View
: Displays the content of a selected file.

Remove
: Cancel the transmission of the selected forecast, waiting in the pending queue, and actually deletes the selected file.

Retransmit

Schedules a product for retransmission. For TAFs, the issue, valid and header times are *not* updated. This action actually moves the selected file from the directory to the `xmit/pending` directory and updates the requested transmission time (as part of the filename).

Help

Displays short help in a text window.

Note

If you want to make changes to the forecast just reload it into the TAF editor. If the forecast is present in the pending queue, the editor will use that instead of using the latest issued TAF.

The maximum amount of time the TAF product can remain in the pending queue is a configurable item in the AvnFPS transmission server's configuration file. If this length of time is exceeded, the pending item is moved to the 'bad' directory. For more details on both of these features, see Section 4, “Forecast Transmission” [93].

2.4. Menu Bar

File

File Menu

Check Now (**Alt-f-c**)
Normally the checks are done upon notification from **avnserver** and every 30 minutes, to check FM and TEMPO groups[1]. This option allows the user to force an immediate check of all TAFs versus current observations and guidance sources.

Restart (**Alt-f-e**)
This option shuts down the current instance of AvnFPS and restarts it. The restart option preserves the selection of monitored product(s). You also have a choice to run AvnFPS under a different forecaster's configuration.

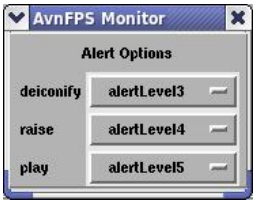
Quit (**Alt-f-q**)
Closes the application.

Options

Options Menu

Setup (**Alt-o-s**)
Can be used to edit forecaster specific configuration file `/etc/app-resources/X.N` where *N* is the forecaster ID. See resource editor in the following section for more details.

Alert (**Alt-o-a**)
Used to select and temporarily override the user's alert criteria when the monitor detects a condition requiring forecaster's action. Selecting the **Alert** option pops up a dialog in which you may select the minimal severity level, including disabled, for triggering a behavior.



Alert Dialog

Blink (**Alt-o-b**)
If selected, the TAF ID button/label will blink on new notification about changing weather conditions affecting the TAF

Help

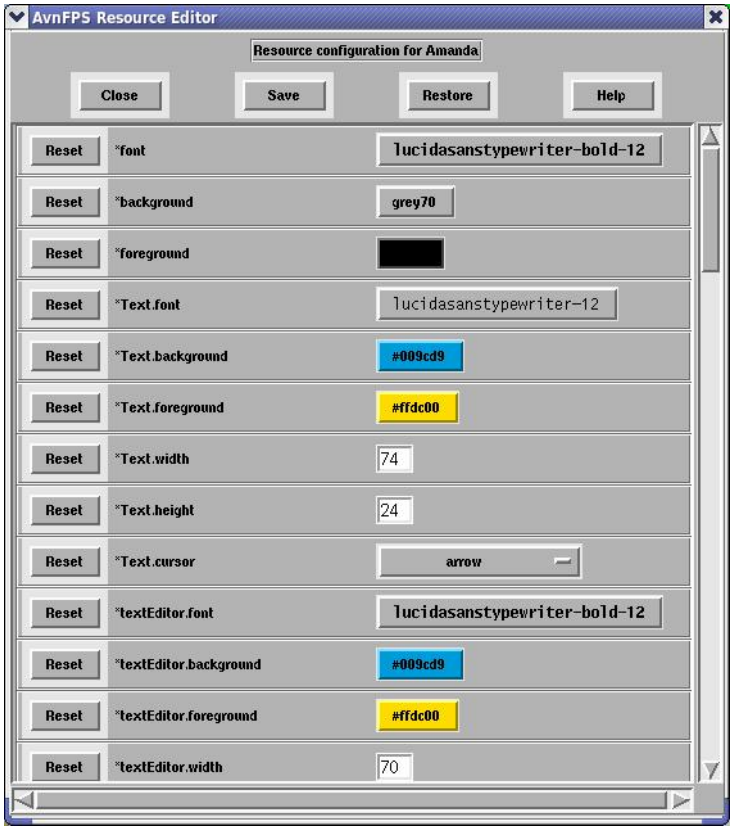
Help Menu

About (**Alt-h-a**)
Provides version number, and contact information.

Usage (**Alt-h-u**)
Displays online help window.

2.5. Resource Editor

Selecting **Setup** from under the Options pull-down menu brings up a new dialog which allows forecaster-specific customization of the AvnWatch and TAF Editor/Viewer GUIs.



Resource Editor Dialog

Within the scrolled window in this editor, there is a list of configuration resources that can be changed by the forecaster. Each frame contains name of the resource. You can display a short description in a balloon pop-up window by pointing mouse cursor at the name. Value of the resource is displayed on the right. Each resource has a 'Reset' button. If pressed, this will return that resource to the AWIPS baseline value. The 'Restore' button at the top of the Editor, resets *all* resources listed to baseline values.

Depending on the resource type it is:

Toggle Button
Used for Boolean values: yes or no (1 or 0 in the configuration file).

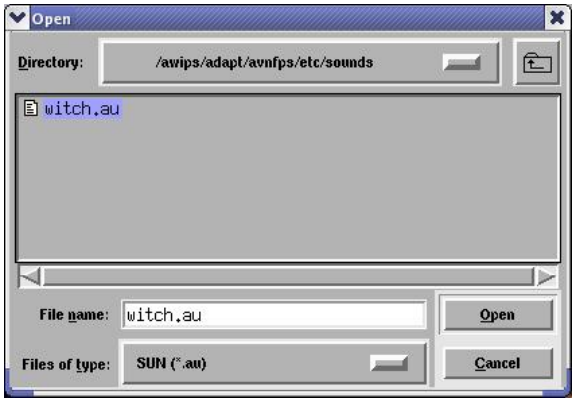
Option Menu
Allows selection from predefined values.

Entry
Allows to type in values, such as width and height

Button
Invokes dialogues specific for a given resource. This can be:

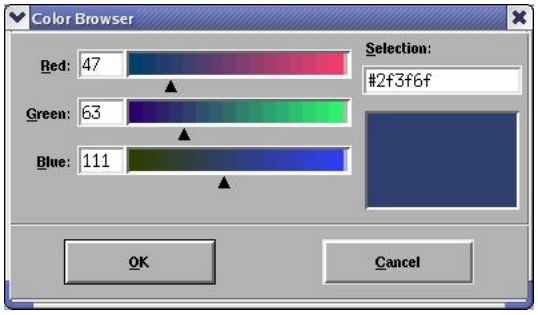
File Selection Dialog

Used to select audio file for alerts, resource `*playFile`



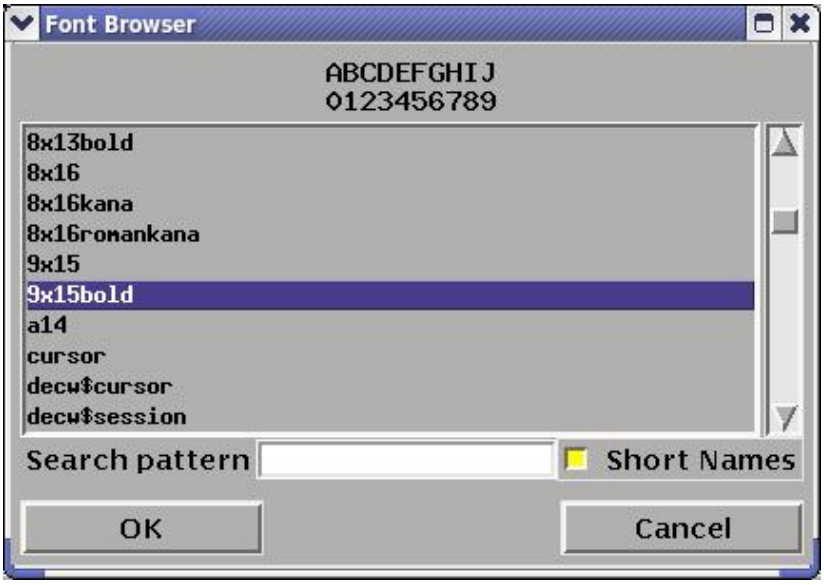
File Selection

Color Chooser Dialog
Alows to set various colors.



Color Chooser

Font Chooser Dialog
Selects fonts from a list of all fonts available on the system.



Font Chooser

The toggle **Short Names** resticts the list to font aliases. This significantly shortens the length of displayed fonts

and is generally sufficient.

Note

When viewing guidance, the data will be displayed most effectively using fixed-width fonts. The title "ABCD...1234" in the Font Browser will aid you in identifying these types of fonts.

2.6. Editor and Product Selection Buttons

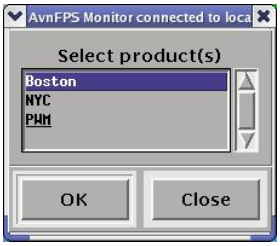
Just below the AvnWatch GUI menu bar there are four buttons.

TAF Editor
This button invokes the Taf Editor. See here for more details Section 3, “TAF Editor” [79].

Climate
This button invokes Climate Menu GUI, allowing forecaster to see the climatological data in a variety of formats. See here for more details Section 6.1, “Climatology Startup Menu” [97].

Plot
This button displays TAFs, observations and guidance in graphical, time-series format. See here for more details Section 5, “Weather Plot” [94]

Backup
The **Backup** button can used to monitor a different set of TAFs on-the-fly. This can be useful when backing up other WFOs' TAFs. AvnFPS, if properly configured, can monitor dozens of TAFs simultaneously. When you select the **Backup** button, the Product Selection Dialog launches:



Product Selection

Tip

If you want to select more than one item from the list, either press and drag the mouse (this works for consecutive items), or click at the items while pressing down the **Ctrl** key.

3. TAF Editor

The TAF editor is a text editor with all the standard capabilities such as cut, copy, paste, find and replace, auto backup. Forecast specific functionality involves syntax checking, climate and current weather consistency, and if configured, airport impact checks. It can also be invoked as a forecast viewer, where inconsistencies between forecast and the recent observations are highlighted.

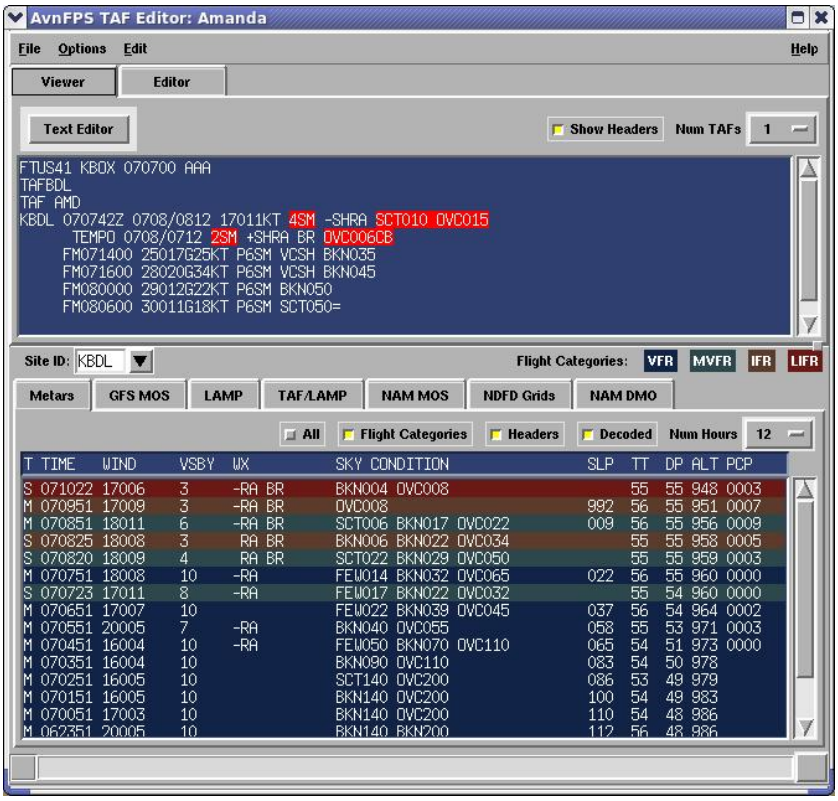
3.1. Starting the TAF Editor

The editor is invoked from the TAF Monitoring GUI. There are two ways to do it.

- Pressing the **TAF Editor** button starts the editor without loading any forecasts. Normally, one would use this method when preparing the routinely-issued TAFs.
- Use one of the Editor Shortcuts for the selected TAF on the right-hand site of the AvnWatch GUI window. This will display the editor in the edit mode, with the most recent, valid TAF loaded for you.

The upper portion of the TAF Editor can be used to either view, "read-only", or edit the TAFs. Two "recipe tabs" near the top of the editor allow you to choose between these two modes. The next section first describes the "read-only", viewer, mode. In following section after that, the full-featured editor is described.

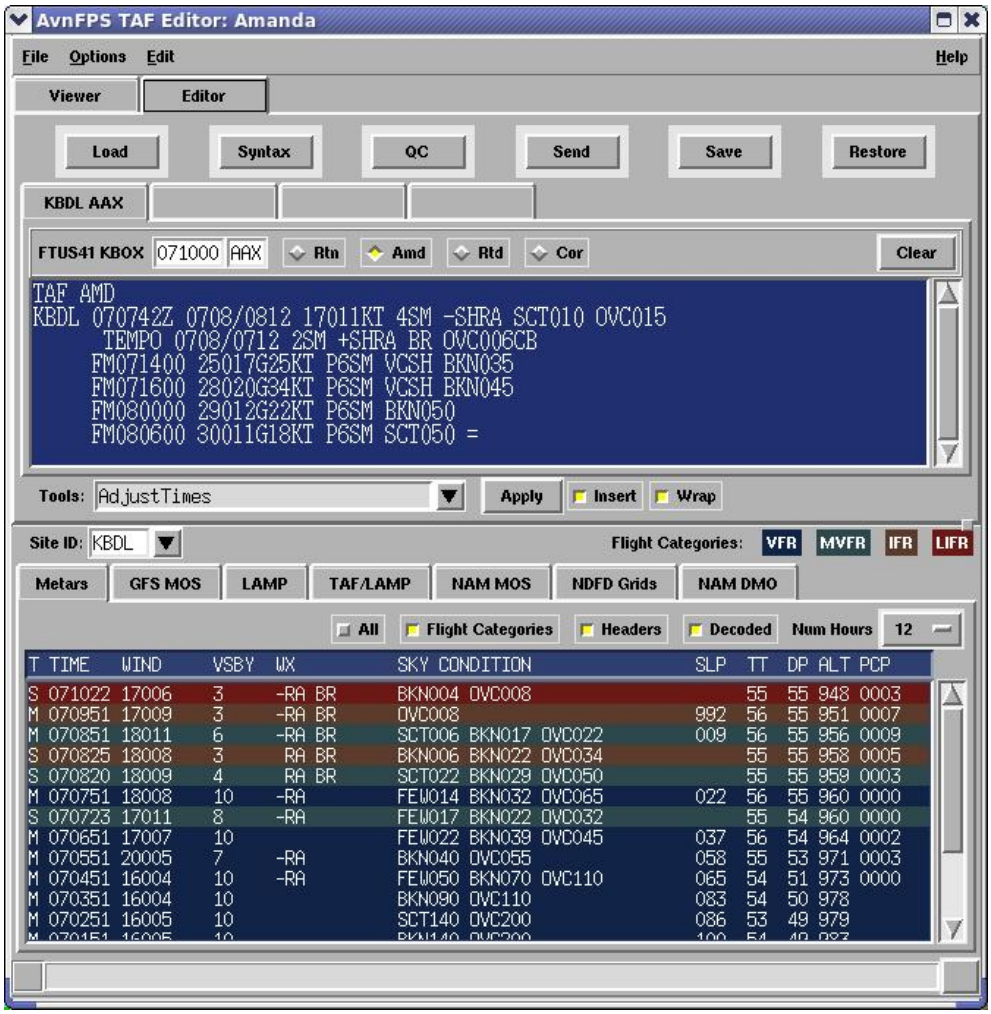
3.2. Using the TAF Viewer



TAF Editor - Viewer Mode

The TAF Viewer Window shows one or more TAF products. If the lower portion of the TAF Editor is set to show METARs, and the TAF disagrees with observations for that airport, then portions of the TAF and the observation are highlighted to show the discreptancy. The **Text Editor** button will change the window to the TAF Editor Window and load the most recent TAF into a new tab. The **Show Headers** toggle will add/remove header information from the TAF products currently in the viewer. The menu labeled **Num TAFs** allows you to choose the number of TAF product versions that will appear in the viewer. The **Site ID** pull-down just below the viewer allows you to choose which TAFs will appear in the viewer.

3.3. Using the TAF Editor



TAF Editor - Edit Mode

When the **Editor** recipe tab near the top of the TAF Editor is selected, the upper portion of GUI becomes a specialized TAF editor. Near the top of the Editor Window is a row of command buttons that perform several functions that can aid TAF preparation. Below the command buttons is a row of recipe tabs that allow you to edit a number of TAFs simultaneously. The text-editing window occupies roughly the upper half of the the GUI. Below the text window are options for running TAF Tools. TAF Tools are user defined functions that manipulate forecasts loaded in the TAF Editor.

To edit the text of a TAF that is loaded in the TAF Editor Window, click within the window. An insert cursor will appear at the point where you click. Edit the text as you would with any similar sane text editing application, except vi. Copy/paste functions are available from the Edit menu and from a right-click menu within the window. Search/replace functions are available from the Edit menu. Header information can be changed by using the menu items above the editing window. The current recipe tab can be cleared by selecting the **Clear** button.

Information on using the Command Buttons and TAF Tools can be found in the following sections.

3.3.1. Command Buttons

Load

Invokes the forecast selection dialogue.

Forecast Selection Dialog

To load TAFs in an TAF Editor window:

1. Select one or more products from the Products menu.
2. Select one or more sites from the Site menu.
3. Select an option from the **Initialize From** menu.
4. Select a **Forecast Type**.

Tip

If you want to select more than one item from the **product** or **Sites** list, either press and drag the mouse (this works for consecutive items), or click at the items while pressing the **Ctrl** key.

Options for initializing a TAF include the following:

latest	Use the most recent TAF otherwise use the template. Most useful for amended and corrected forecasts.
template	Use the template. Most useful for preparing routine forecasts.
merge	Merges the latest TAF and the template. Most useful for sites with scheduled part time observations.

Important

If a forecast is waiting for transmission in `/awips/adapt/avnfps/OB9.2/xmit/pending` directory, it will be loaded into editor, regardless of the 'Initialize From' menu choice and then deleted from the pending queue.

Syntax

Syntax stands for syntax quality control. When you press this button, the editor will attempt to decode all forecasts and reassemble them, assuring proper indentation and maximum line length. Any errors/warnings will be highlighted in the displayed bulletin. There are 3 levels of errors:

- Red - if the decoder cannot determine the meaning of particular word. This is a fatal error, the rest of the forecast (for a given site) is not decoded.
- Orange - this indicates an error as specified by NWSI 10-813 (shall not ...).

- Green - this indicates a warning as specified by NWSI 10-813 (should not ...).

Positioning the mouse pointer over the highlighted text will cause an explanatory message to appear.

Syntax QC

QC

The **QC** combines several checks. Currently implemented are: Current Weather Check, Climate Quality Control and Impact Check. When clicking on the **QC** button using the left-mouse click, the QC settings for that particular TAF Section 4.2, “Editing TAF Site Information” [35] will be performed.

Important

The QC check is *optional*. Any highlighted text resulting from this action does not necessarily indicate that the TAF(s) is incorrect. Results from these checks merely require an evaluation “Are you sure?” before dissemination and not necessarily demand action on the part of the forecaster.

To override the default settings for the TAFs shown in the editor text window, right clicking on the **QC** button will bring up the following dialog.

QC Selection Dialog

By selecting or deselecting the options provided, and then pressing the **Apply** button, this will temporarily override the default QC checks performed for each TAF.

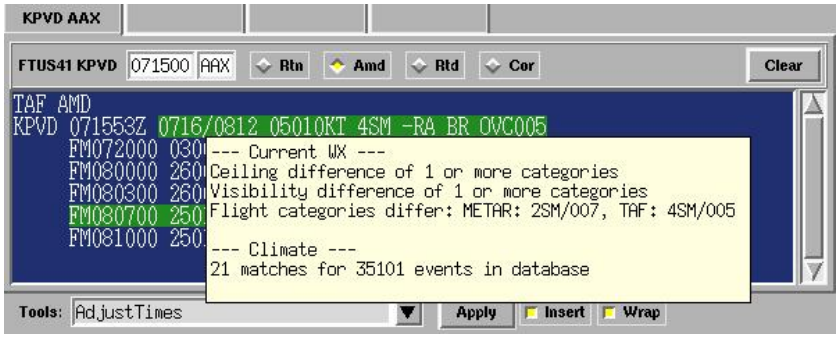
The Current Wx check is intended for rapidly changing conditions during forecast preparation, or when a routinely issued forecast is prepared well in advance of the transmission window. If current observation and the the first line of the forecast do not pass all of the monitoring checks, the first line of the TAF will be highlighted.

The Climate Quality Control check assesses the climatological frequency of the weather element combinations found in the TAF. It scans the climatological dataset of hourly and special observations, comparing combinations found in the TAF with observations available for that station. If a combination in the TAF has a low climatological frequency, an alert is displayed on the screen, alerting the user to a forecast of a climatologically rare event.
See Appendix C, *Climate Quality Control* [] for detailed description of the algorithm.

The Impact Check examines each line of a TAF against airport specific conditions, such as low ceilings and visibility or significant crosswinds as these may affect aircraft arrival and departure rates. The impact rules can (and should) be customized for each airport.

See the section called “Impact QC Definition File” [19] for detailed description of the configuration file and

how to customize it for each airport.



Current Wx and Climate Output

Save

Stores current TAF bulletin into a temporary file *`tmp/CCCWRKTAF.type`*, where *`CCC`* is your node id and *`type`* is the forecast type.

Restore

Restores bulletin previously stored in a temporary file *`tmp/CCCWRKTAF.type`*. *`type`* must match the type selected in the editor.

Send

Puts forecast in the transmission queue. In order to succeed, the forecast must successfully pass the Syntax check. In your resource file, there is an item *disalowSend, by default set to error. This means you can't inadverently send a forecast which does not follow the directive. If you have an error flagged, but are certain that the forecast is correct and either the software or directive is wrong, use **Clear Errors** before pressing **Send**. A confirmation dialog will be shown, where you will have to choose your name from the list. The corresponding forecaster ID number will be logged for forecast verification purposes.

Send Dialog

You can manually set the transmission time of the product. This feature is provided to allow for sending routine forecasts closer or at the end of transmission window in case of rapidly changing weather conditions.

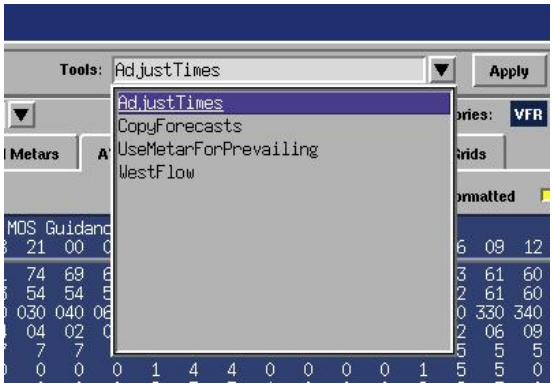
Note

The transmit flag in the `etc/forecasters` file controls whether this button appears or not in the TAF editor. This feature is useful if users wish to use AvnFPS without the risk of inadvertant transmission of a TAF.

3.3.2. TAF Tools

TAF Tools are user-defined functions that manipulate forecasts loaded in the TAF Editor. Additional TAF Tools can be developed and implemented by programmers at your WFO, much like the locally-developed Smart Tools that can edit grids in the Graphical Forecast Editor. TAF Tools are written in Python, just like Smart Tools. TAF Tools can access just about any data source that appears in AvnFPS. This includes the TAF in the TAF Editor, other TAFs, METARs, and various forms of guidance.

A small set of TAF Tools are delivered with AvnFPS. These tools were developed for their instructional value as well as their utility. They are described below:



TAF Tools

Adjust Times

Modifies TAF by removing past periods, adjusts issue and valid times. If you use this tool before running syntax quality control, you can avoid error messages about invalid forecast periods.

Copy Forecasts

Allows for interactive copy of forecasts in a collective. The idea is that you prepare forecast for one site, and want to replicate weather conditions for some other TAF sites nearby. This tool can be an alternative for copy-and-paste.

Use METAR For Prevailing

Updates forecasts with the current observations. Also incorporates functionality of the **Adjust Times** tool.

West Flow

An example of a more sophisticated tool; it addresses a situation that commonly occurs in West Virginia. The tool creates forecasts for a number of other sites based on forecasts for KPKB and KHTS. The tool advects clouds and weather, adjusting cloud heights for elevation. Since the tool specifically access the WV TAF sites by name, it cannot be used “as is” by other WFOs.

3.4. Using the Guidance Viewer

The Guidance Viewer Window allows you to view various forms of guidance as you compose TAFs. Currently, AvnFPS can be configured to display up to eight data sources. These are Metars, GFS MOS, NAM MOS, LAMP, TAF/LAMP, NAM DMO (Direct Model Output), NDFD Grids, . Each source has a dedicated display window which you select by clicking its tab.

All viewers have toggle **All**, if set, will display data for all sites associated with the currently monitored product. The toggle **Flight categories** will highlight periods according to the observed (METARs) and forecasted weather. Most of the guidance available in the TAF Editor can displayed in one of the 3 formats, depending on the toggle **Format**:

table	The display is similar to that of MOS text messages, with added optional display of probabilities for VIS and CIG categories (the MOS text bulletins contain only the best category).
long	The data is formatted as a TAF. There is one line per each forecast period (1 or 3 hours).
short	The data is formatted as a TAF. Forecast periods with similar weather are combined.

The toggling **Flight Categories** button will higlight each forecast period/line depending on the flight category.

The algorithm to combine periods with similar weather is based on COMET [http://meted.ucar.edu/dlac/lesson3b] training module. Major rules applied are:

- Flight category: time periods are not combined in one TAF line unless it is configured by the users to merge by using `grp_taf.cfg`.
- Weather and wind condition changes: when there is a change (starting or ending) in weather conditions and wind changes, new line of TAF is always formed.
- Combining rules: Rules used to pack TAFs into more concise form. These rules are site configurable and are stored in `grp_taf.cfg`.

Metars

Displays METARs for a selected site. You can select the site from the **Site ID** list. The items on the list are set to the currently monitored product(s).

METAR Viewer

The number of displayed reports is controlled by the **Num Hours** menu. If there is a discrepancy between the TAF and the most recent METAR, the offending weather element will be highlighted. There are two display modes: text and decoded. In the text mode you may switch the display of WMO headers. In decoded mode there is an option to highlight reports according to the flight category. The toggle **Headers** is to display WMO headers. The toggle **All**, if set, will display all METARs for the selected product(s). It replaces **All METARs** found in the previous releases of AvnFPS.

GFS MOS Viewer

Displays GFS MOS guidance for a selected site. You can select the site from the **Site ID** list.

GFS MOS Viewer

The toggle **Routine** button switches TAF valid period between amending the current TAF and the next cardial issue -- easier to see, rather than explain. The **Probabilities** button displays the probability of the categorical elements in the product. These probabilistic data are not available in the MOS *text* products, but rather are disseminated in the BUFR bulletins.

NAM MOS Viewer

Displays NAM MOS guidance for a selected site. The display format and control options are the same as for GFS MOS.

LAMP Viewer

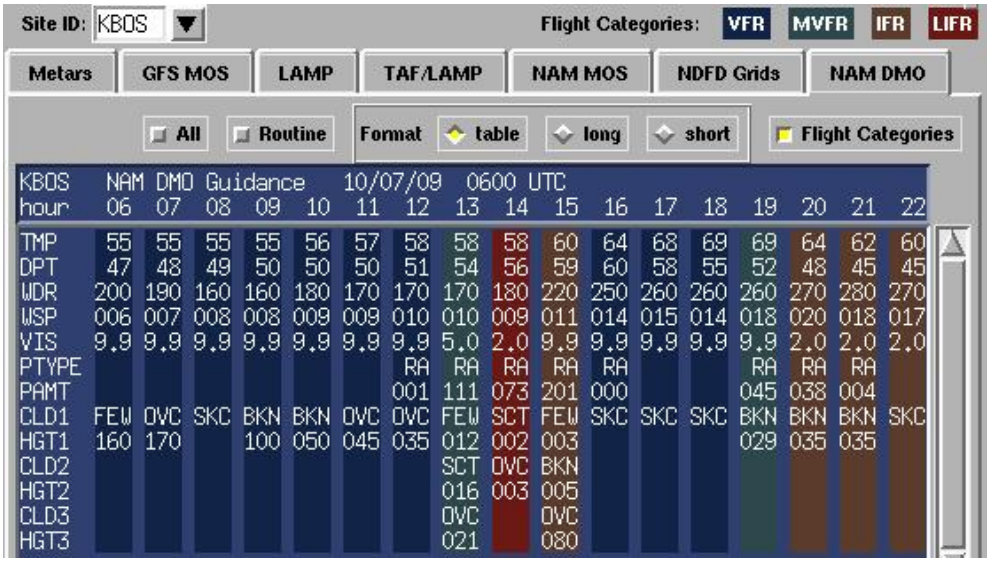
Displays LAMP guidance for the selected site. LAMP is based on GFS MOS and provides updates every hour using surface observations, satellite and lightning data. [LAMP]

TAF/LAMP Viewer

Displays the official TAF updated with LAMP guidance. Since this is a synthesis of LAMP guidance and the Official TAF, it's not meaningful to have a 'table' format, just 'short' and 'long' TAF formats are available. [Oberfield, et. al.]

NAM DMO (Direct Model Output) Viewer

Displays minimally processed output from the NAM model for a selected site.



NAM DMO Viewer

The display is similar as for GFS MOS guidance. There is no **Probabilities** button as this is data taken directly from a single deterministic run of the NAM. Visibility and cloud ceilings are not state variables in the model but are derived according to algorithms described by [Smirnova, et. al.]

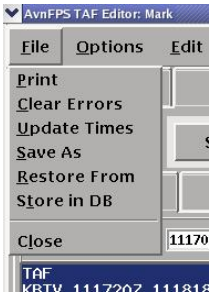
NDFD Grids Viewer

Displays data taken from NDFD grids for the gridpoint closest to the selected site. The display format and control options are similar to those for the GFS MOS Viewer, with the exception of **Probabilities** button.

3.5. TAF Editor Menu Bar

The Menu Bar allows you to access controls for many features of the TAF Editor. Entries in parentheses indicate keyboard shortcuts that will access that menu item.

File



File Menu

Print (**Alt-f-p**)
Launches the print dialog. The dialog displays the default print command: **lpr** and print on the default printer for your workstation. If you want to use a different printer, enter appropriate options (for example: **`lpr -Plp2`**).

Tip

You can use this dialog to add content of the text window to fxatext database: enter **`textdb -w`** ***`CCCNNNXXX`*** with the appropriate AFOS PIL.

Clear Errors (**Alt-f-c**)
Clears error tags set by Syntax and QC actions. It can also be used to force transmission of forecasts that did not pass Syntax checks. Thus, the forecaster is never prevented from issuing the forecasts.

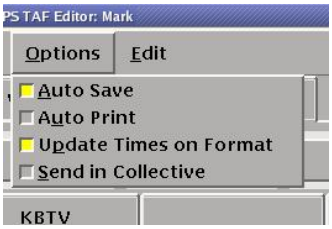
Update Times (**Alt-f-u**)
Updates issue and valid times in all forecasts loaded in the TAF Editor.

Save As (**Alt-f-s**)
Allows users to save edited forecast to a file. Invokes the file selection dialog.

Restore From (**Alt-f-r**)
Allows users to restore forecast from a timed-backup file. Invokes the file selection dialog.

Store in DB (**Alt-f-t**)
Stores current TAF bulletin to fxatext database, under the AFOS PIL *`CCCWRKTAF`*, where *`CCC`* is your node id.

Options



Options Menu

Auto Save (**Alt-o-a**)
When selected, the content of the window is saved every 60 seconds to a backup file. If needed, the file can be restored by using the Files->**Restore From** (**Alt-f-r**) The directory where the backup file is stored is `/awips/adapt/avnfps/OB9.2/tmp`. The file is named taf.*`host`*, (for example taf.lx3). When a bulletin is reloaded, the file is renamed to taf.*`host`*.prev.

Auto Print (**Alt-o-u**)
When selected, the forecast will be printed on the default printer when the **Send** button is pressed.

Update Times on Format (**Alt-o-p**)
When selected, issue and valid times in all forecasts will be updated (with values depending on the forecast type) when the **SynQC** button is pressed.

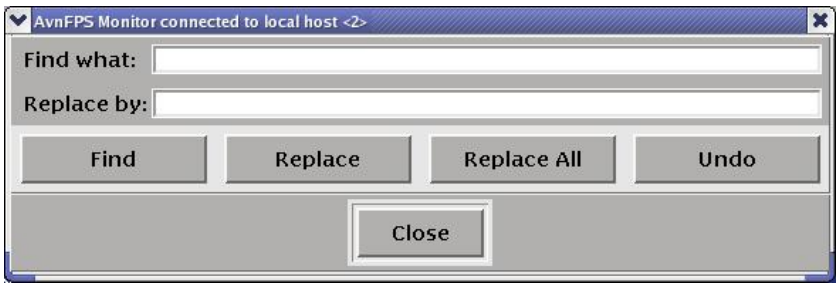
Send in Collective (**Alt-o-s**)
Normally each TAF is sent separately. When this option is selected, the contents of the text window, perhaps consisting of one or more TAFs is transmitted as one file. This option is used by OCONUS sites. Routine use of this option is discouraged

Edit

Edit Menu

Cut (**Alt-e-c**) , **Copy** (**Alt-e-o**) , **Paste** (**Alt-e-p**) , **Undo** (**Alt-e-u**) , **Redo** (**Alt-e-r**) ,
These options provide standard editing capabilities. Undo and Redo options have very large histories.

Find (**Alt-e-f**)



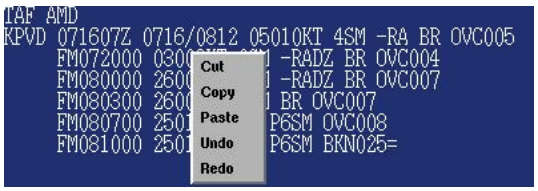
Find and Replace Dialog

This option invokes find and replace dialog with undo capability.

Tip

By pressing the right mouse button in the text area, you will get a popup menu, which also

provides editing capabilities.



Editor Popup Menu

Help

Key Bindings (**Alt-h-k**)
Displays key bindings valid in the text editor window. These are standard bindings as implemented in Tk. For complete list, see Appendix 1.

Usage (**Alt-h-u**)
Displays on-line help.

4. Forecast Transmission

Forecasts made within AvnFPS are sent by a dedicated transmission server. When you press the **OK** button in the **Send** [85] Dialog, the contents of the forecast editor is placed in the transmission queue (that is, written to a file in the directory `xmit/pending`). AvnFPS transmission server checks the queue periodically. When the transmission time of a product is reached, the forecast is disseminated. The transmission server will then notify all instances of AvnWatch whether the transmission succeeded. The Monitoring GUI's **Queue** button will flash and the results of the transmission will be displayed in the Status Bar at the bottom of the Monitor. If the transmission fails, the **Queue** button will change its color from green to red. You can view the status by clicking on the button. This will display the Transmission Queue Dialog, see Section 2.3, “Transmission Queue” [71]. To retransmit a forecast, select the **bad** option, highlight the failed entry (listed in the bottom window), and press **Retransmit**. The forecast will be moved back to the pending queue and the transmission server will attempt to send it again.

Note

The transmission server does not actually transmit the forecast. It simply invokes AWIPS utility **handleOUP.pl** and checks for the return status. If the forecasts are not transmitted, one should check **handleOUP.pl** log file, on the system where the server runs. Most likely it will be `px2`.

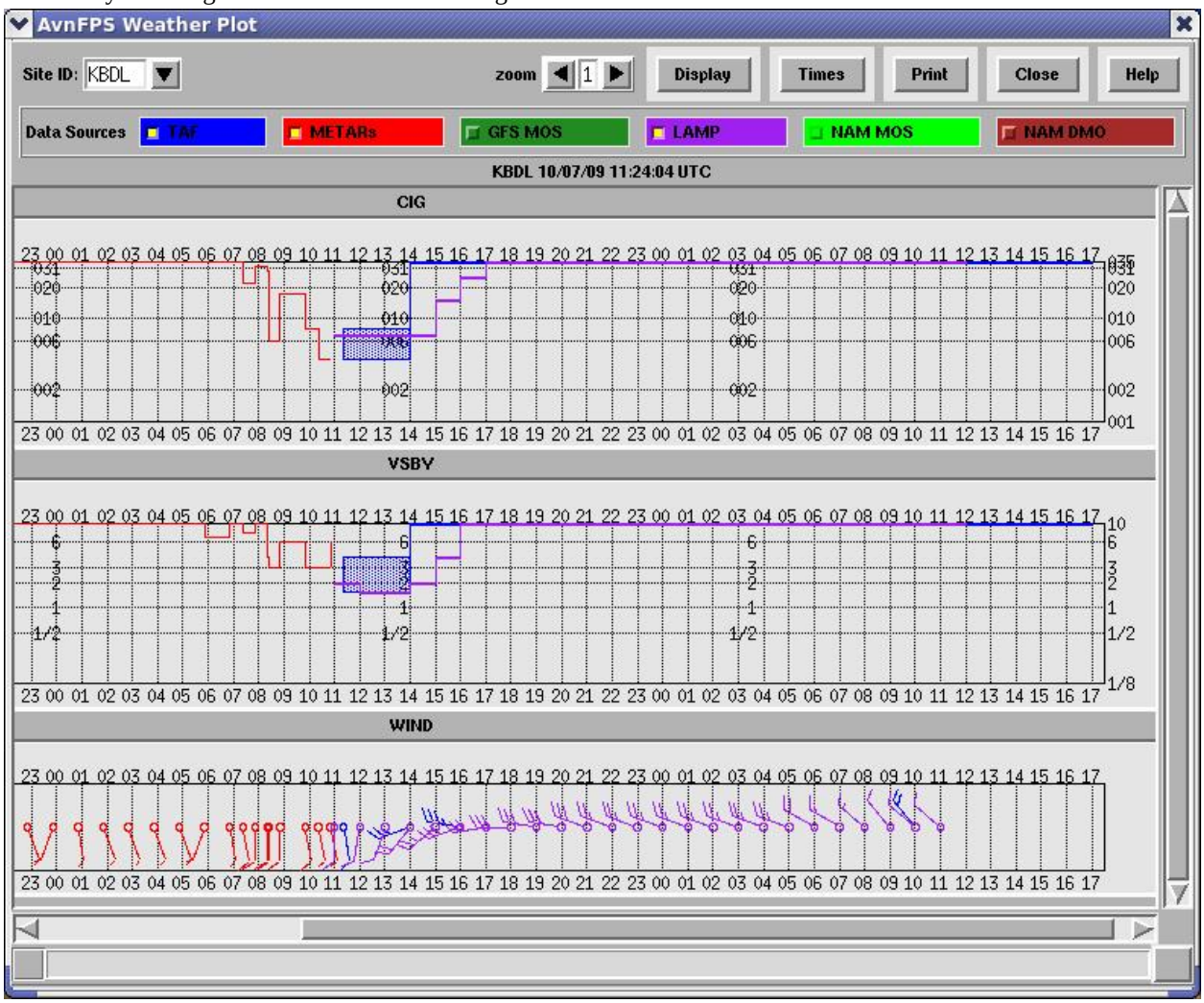
Important

If you scheduled a forecast to be transmitted at a future time, you can easily pull it from the queue if changing weather condition warrant its modification. Any time you load a forecast into the editor, it checks whether there is a similar TAF waiting in the pending queue. If so, it is pulled out from the queue and loaded into the editor.

The transmission server collects information on forecasts sent by the WFO for verification purposes. A product with WMO ID header NXUS98 *`CCCC`*, where *`CCCC`* is your site id, is sent every 6 hours over the WAN where its collected as part of the Stats-On-Demand service.

5. Weather Plot

Weather Plot displays METARs, guidance and current and past forecasts in the form of a time series. This dialog is invoked by selecting **Plot** in the TAF Monitoring GUI.



Weather Plot

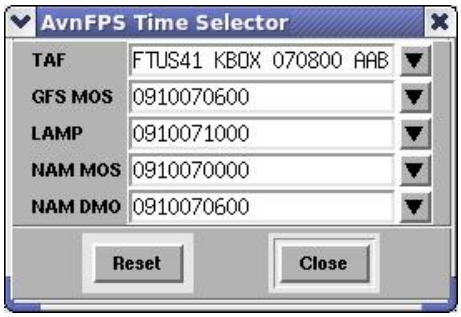
Ceiling, visibility and winds are plotted as separate graphs. The horizontal axis in each graph represents time, the current time approximately centered in the display. The number of hours past and in the future is configurable, up to a point. The ticks on the vertical axes for ceiling and visibility are the significant thresholds defined separately for each TAF site.

For each forecast period, ceiling and visibilities are represented by a tuple (low, high) values, and plotted as a shaded rectangle. Thus such a rectangle indicates existence of a TEMPO or PROB30 group. Only prevailing winds are plotted.

To plot data for a site, select the TAF id from the **Site ID** menu. This will display all data selected in the **Data Sources** area. Currently available data sources are: TAFs, METARs, NAM MOS, GFS MOS, and NAM DMO, . Depending on the configuration at your WFO, only a subset may visible. However the **TAF** and **METARs** buttons should always be present. By default, the most recent data is displayed.

If you want to plot previous forecast or guidance, select the **Times** button. A dialog displaying available data will be

shown.



Time Selector

Click on the arrow on the right to show all available issuance times, select desired time, then press the **Display** in the main window. The **Reset** button resets all choices to the most recent time.

During changing weather conditions, the reporting frequency of surface observations may be high, making the display, particularily wind plot, very "busy". In such case you may want to zoom the display, from the **zoom** menu select zoom factor, then press the **Display**. This button redraws the display and may be used when new observations are received, as the display is not automatically refreshed.

To save the graphs, use the **Print** button. By default, this will create a JPEG image in the `/awips/adapt/avnfps/OB9.2/tmp` directory. The WxPlot's **Print** command to run is located in the `etc/wxplot.cfg` file under the [print] tag. The '%s' wildcard--expanded to the identifier in the **Site ID** text field--in the command line string is optional. Thus, if you wanted the **Print** button to send the image directly to the printer, the command line string could be changed to this:

```
[print]
cmd=convert - -size 600x1000 jpg:- | lpr -o landscape
```

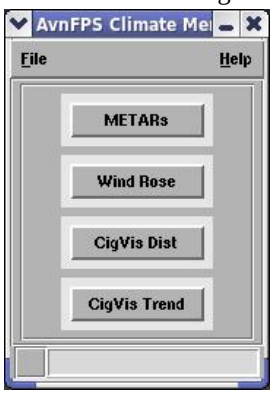
See Section 2.1, “Files in `etc`” [11] and refer to the `etc/wxplot.cfg` section for more details.

6. Climatology Tools

Beginning with AvnFPS 3.0, a climatology database was introduced as another guidance source to help forecasters prepare and QC their TAFs. The climatology database comprise of flat files, one for each TAF site. These files consist of NCDC yearly archives of surface observations, reformatted for fast indexing and retrieval. All of the GUIs can be started as standalone programs, or from the AvnFPS Startup Menu, see Section 1, “Startup Menu” [], or from the **Climate** button on the AvnFPS Main Monitoring GUI.

6.1. Climatology Startup Menu

To start any of the climate tools, select the **Climate** button in the AvnFPS Startup Menu, see Section 1, “Startup Menu” []. The Climatology Startup Menu, contains two items. Select the **METARs** button to launch the Historical METAR Tool, and select the **Stats** button to launch the Climatological Statistics Tool.



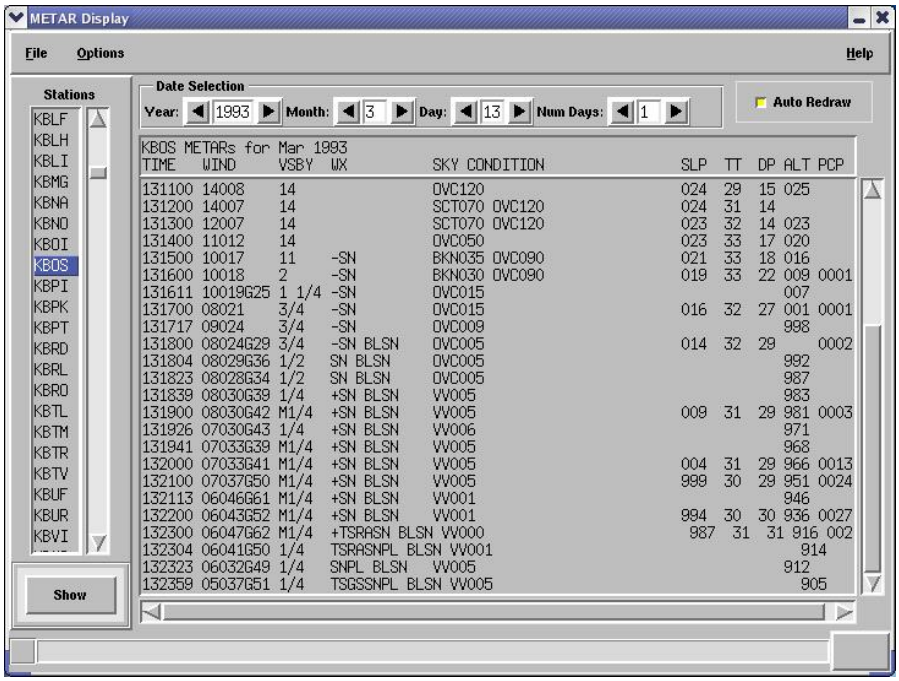
Climate Selection Menu

6.2. METAR Display

The Historical METAR Tool reconstructs hourly and special weather reports from the climatological database and renders them in the form of METARs. Each report contains as many of the following weather elements as are available: wind, visibility, significant weather, temperature, dew point, and 6- and 24-hr precipitation amount.

Important

The records in the database are merged from several data sources and the first records predate the era of METARs in the US. Thus the reports displayed are not true METARs.



METAR Viewer

To view observations for a particular day:

1. Select the desired station from the **Stations** list on the left side of the dialog.
2. Use the **Y**, **M**, **D** spinners to set the desired year, month, and day.
3. Select the **Show** button.

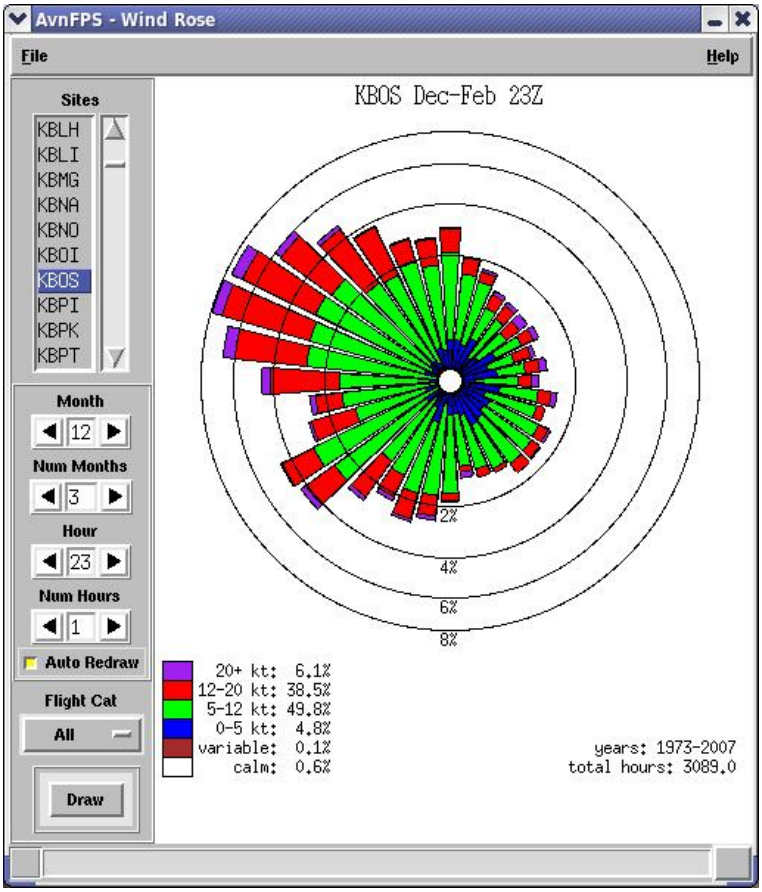
The window will be updated with the appropriate observations.

To print the set of observations:

1. Select the **Print** button.

6.3. Wind Rose

This GUI displays a wind rose for a given month (or a range of months) and hour (or a range of hours). You can also specify flight category to further refine the results.



Wind Rose

To display wind rose:

- Use arrows to set month in the **Month** counter. If you want statistics for a range of months, use the arrows surrounding the window labelled **Num Months** label to increment/decrement the value in the text window.
- Use the arrows in the **Hour** counter to select an hour. If you want statistics for a range of hours, use the arrows surrounding the window labelled **Num Hours** label to increment/decrement the value in the text window.
- Use the **Sites** list to pick the location of interest.
- Press the **Draw** button to draw the wind rose.

Note

Given selected month, or range of months, hour, or hour range, the **total hours** display is the sum of all the time intervals that match the criteria specified above.

Note

If the **Auto Redraw** button is enabled, then changing any of the parameters listed above will force a redrawing of the results.

The displayed data can be saved to a file as text or graphic and can be printed. those features are accessible through the Menu Bar.

File Menu

To save statistics to a text file:

- Select the **Save Stats** menu item.
- Type the file name in the popup dialog, then press **Save**.

An example output file:

```
KBUF Jan
HOUR:          00Z
TOTAL:       991.8
CALM:         20.7
SPEED:       0-5 kt   5-12 kt  12-20 kt    20+ kt
VRB:           0.0       0.5       0.0       0.0
349-011:       5.5      16.2       1.5       0.0
011-034:       3.2      25.5       3.5       0.0
034-056:      10.7      32.2       3.9       0.4
056-079:      11.9      49.2      16.5       0.0
079-101:       1.5      21.8       4.0       0.0
101-124:       2.2      14.5       0.0       0.0
124-146:       4.0      17.0       2.1       0.0
146-169:       6.7      29.1       3.5       2.0
169-191:       3.3      35.2      11.1       1.0
191-214:       7.1      22.6      30.4       3.4
214-236:       3.4      42.4      52.7      19.1
236-259:       7.4      59.5      96.3      52.5
259-281:       3.6      41.1      68.6       9.2
281-304:       4.0      20.5      30.7       1.7
304-326:       3.5      16.0       4.1       2.1
326-349:       3.1      21.2       5.3       0.0

HOUR:          01Z
TOTAL:       989.8
CALM:         18.1
SPEED:       0-5 kt   5-12 kt  12-20 kt    20+ kt
VRB:           0.0       0.0       0.0       0.0
349-011:       3.8      14.6       3.0       0.0
011-034:       5.2      13.4       2.8       0.0

                ...
```

To save image to a file:

- Select the **Save Image** menu item.
- Type the file name in the popup dialog. To specify file format, use one of the extensions: .ps, .jpg, .png. Press **Save**.

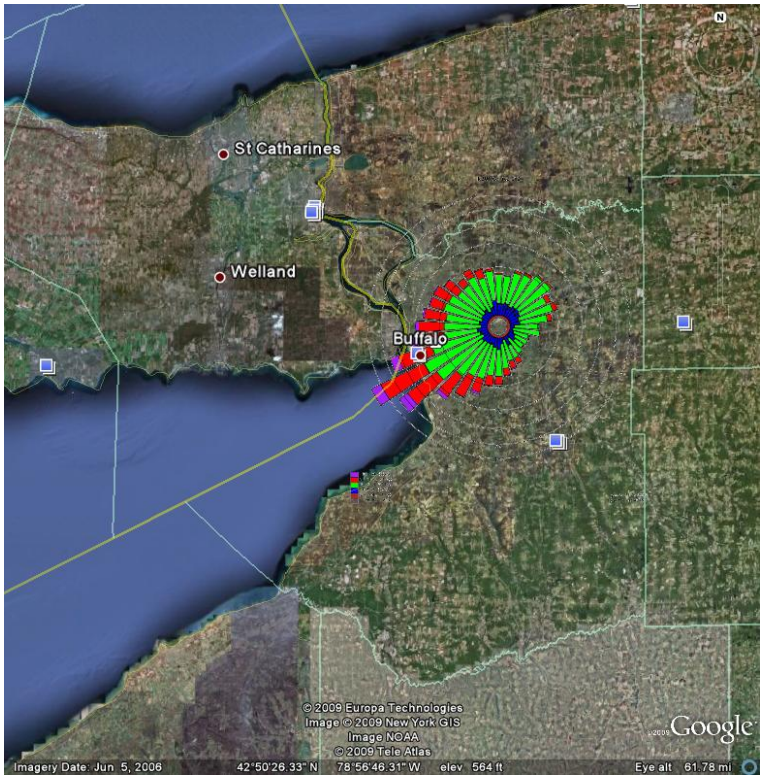
To save an Wind Rose image suitable for displaying in Google Earth:

- Select the **Save for Google Earth** menu item.
- Default name of image is provided but can be overridden in the popup dialog. Google Earth supports only the .png format. Press **Save**.
- When saving the image, an accompanying KML file is produced in the `/awips/adapt/avnfps/OB9.2/tmp` directory along with the image file. You can move the pair of files, to a different host where Google Earth software is installed. Within the KML file, the Icon attribute will need to be edited to include the full path to the PNG image on the new host.
- Using your favorite editor, locate <Icon> tag within the KML file and alter one of the lines within the tag as appropriate for the operating system where Google Earth is installed. Both lines are provided to provide an example of how the information is to be entered, the first line, the default, is for Window systems. The second line, suitable for UNIX based systems, is commented out by the !-- > and -- > tokens. Edit either line as appropriate for your new host and delete the other. Save your changes.

```
<Icon>
  <href>file:///c:/windows/path/to/KBUFOct18-21Z.png</href>
  <!-- href>/linux/path/to/KBUFOct18-21Z.png -->
</Icon>
```

- Once saved, open the KML file in Google Earth: the application's display will position the viewer, and hover over the Wind Rose graphic.

The resulting appearance of the Wind Rose in Google Earth for the Greater Buffalo International Airport (KBUF) showing the pronounced "lake breeze" effect during the mid-afternoon hours for October from 18Z to 21Z:



KBUF Wind Rose in Google Earth

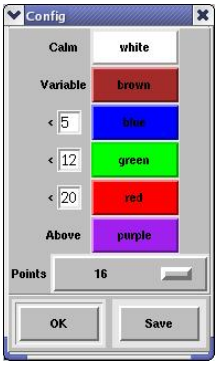
The apparent size of the image in Google Earth and its north-south placement can be adjusted in the WindRose X resource file. The resource file for the wind rose is `/awips/adapt/avnfps/etc/app-resources/XWindRose`. This file can be edited using a text editor. There are 2 pertinent entries for manipulating an image for Google Earth:

- kmlsize: Height (north-south extent) of image (in kilometers) when displayed in Google Earth. The default value is 10 (km).
- kmlsoffset: Proportional image offset in the vertical from image and wind rose centers.

To print image:

- Select the **Print Image** menu item.
- If needed, type in printer destination:
  `lpr -P` *`printer`*
  If the -P option is not specified, the image will be printed on your default printer.
- Press **OK**

To change wind speed thresholds and corresponding colors and to set number of wind directions use the **Configure** menu item. This will display a configuration dialog.



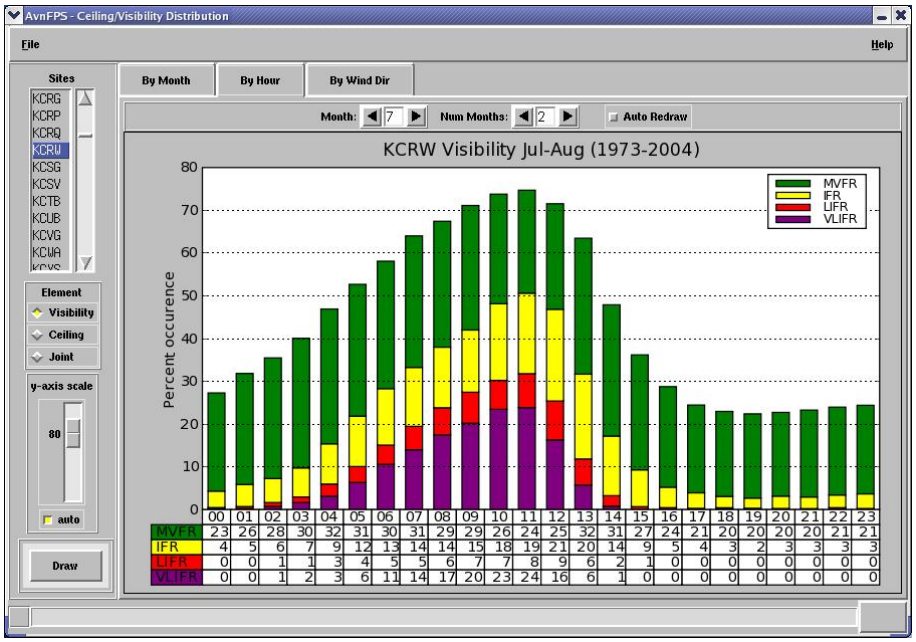
Configuration Dialog

- To change wind speed thresholds, enter appropriate values in the 3 available windows. Number of wind speed categories cannot be changed.
- To change colors, press a button to the right of the wind speed label/entry. A color editor will be shown.
- Use **Points** menu to select number of wind rose compass points, 8, 16 or 36.

Press **OK** to accept the changes. You may save the changes in a file, use **Save**.

6.4. Ceiling and Visibility Distribution

The Ceiling and Visibility Distribution GUI displays distribution of visibility and ceiling by month, by hour, or by wind compass. The display is in a form of a histogram. Image below is hourly flight category distribution according to visibility for the July-August months at Yeager Airport in Charleston, WV. These data show the typical summer-time 'fog season' at this airport with the worst impacts to visibility occurring around sunrise with any LIFR (or worse) conditions dissipating by 14Z.



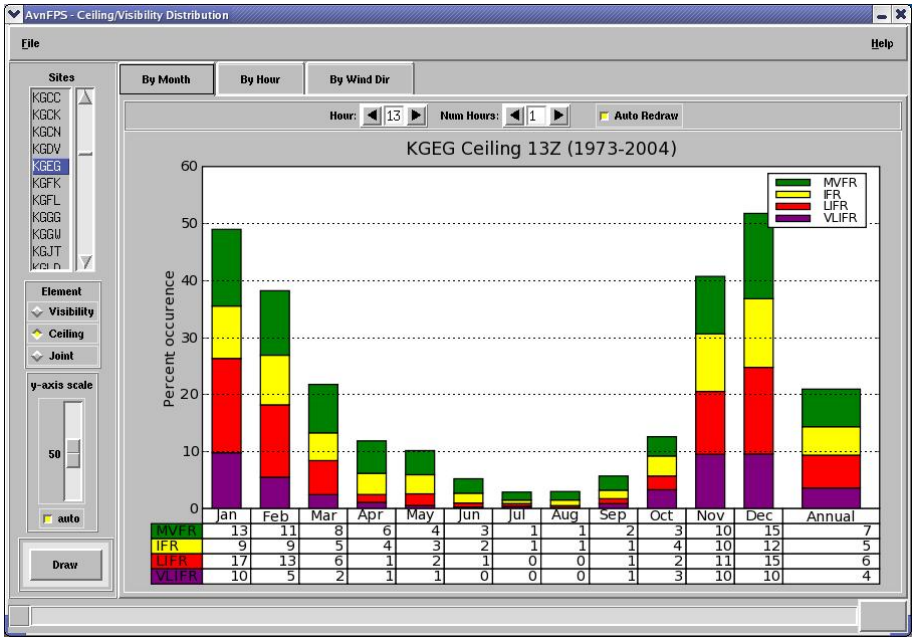
July-August Visibility Distribution by Hour at Yeager Airport

To display hourly distribution:

- Select the **By Hour** tab, above the drawing area.
- In the **Element** frame, select the weather element or, to display flight category, **Joint** toggle.
- Use the arrows surrounding the **Month** text field, just above the drawing area. If you want results for the range of months, increment and decrement the **Num Months** field by using the arrows on either side.
- Use the **Sites** list to pick the locale of interest.
- Press the **Draw** button to draw the histogram.
- There is a slider-bar widget labelled **y-axis scale** which allows you to stretch, or compress, the vertical axis of the plot. The software usually picks a value that makes the most effective use of the drawing canvas. By unselecting the toggle button **auto** below the slider-bar, this feature can be disabled.

To display monthly distribution, proceed as above, but select the **By Month** tab. You can now select and filter results based on a single or range of hours.

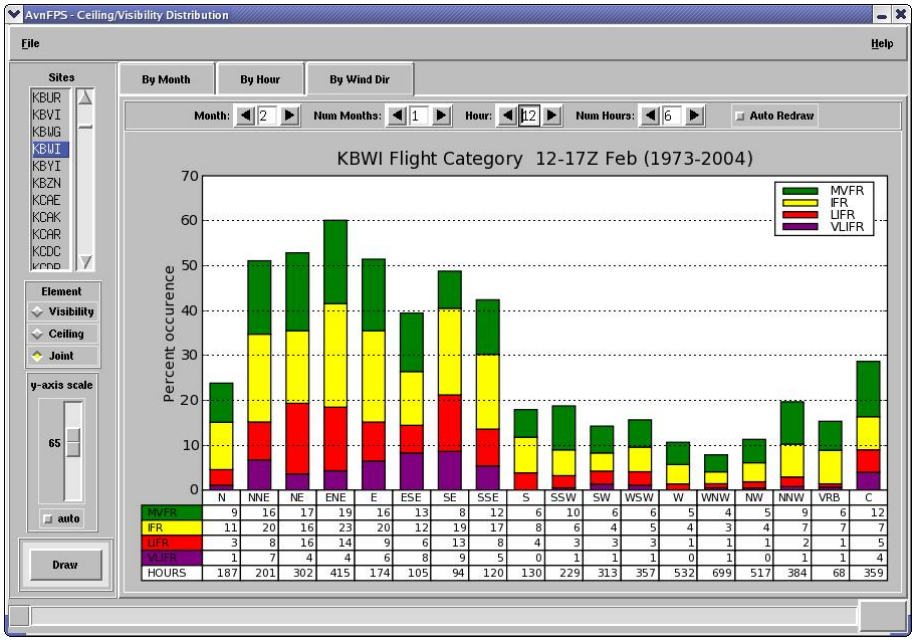
In the image below for Spokane International Airport shows at 13Z, MVFR ceiling conditions, or worse, half of the time during the months of December and January.



Monthly flight category ceiling distribution at Spokane International Airport

To display flight category by wind direction, select the **By Wind Dir** tab. You can select and filter results based on a single or range of hours and month.

In the image below for the Baltimore International Airport, shows in February, for a six hour period from 12Z to 17Z, that MVFR or worse conditions are likely with winds having a easterly fetch. Note the table below with the **HOURS** row indicating the distribution of winds during that time. In this case, west-northwest winds are most common during this period. However in the relatively infrequent situations where winds have an easterly component, there is a significant risk of MVFR or worse conditions during those times.



Flight category distribution by wind direction at Baltimore International Airport

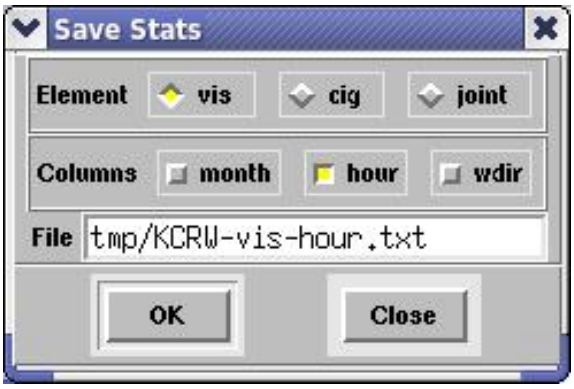
Note

It may take up to a minute to retrieve statistics from climate data file. To speed up display, the toolkit uses a caching mechanism: the data is retrieved only once and then stored in a file, to be read on subsequent displays.

Note

If the **Auto Redraw** button is on, selecting a toggle or changing hour/month will force a redrawing of the results.

The displayed data can be saved to a file as text or graphic and can be printed. Those features are accessible through the Menu Bar. The menu items are the same as for Wind Rose, with the exception of **Configure**. To save statistics to a text file:



Dialog displaying available options for saving statistics to file

- Select the **Save Stats** item. The Save Stats GUI will appear, allowing the selection of various parameters and location and name of the resulting output file.
- Type the file name in the Save Stats dialog, then press the **OK** button.

An example output file:

```
KCRW (1973-2007)
HOUR: 00Z         Visibility
MONTH VLIFR   LIFR    IFR     MVFR    VFR
Jan   8.08    21.37   60.93   93.43   888.75
Feb   2.27    11.47   61.43   85.70   811.18
Mar   2.23    8.12    42.88   76.98   941.35
Apr   1.00    4.00    22.90   50.08   961.12
May   2.32    3.62    36.78   76.57   954.62
Jun   0.55    1.33    19.23   119.98  883.90
Jul   1.93    3.98    34.52   228.52  776.02
Aug   1.55    2.20    43.77   239.45  763.03
Sep   1.37    5.28    38.55   125.60  843.78
Oct   2.30    6.27    43.42   56.82   937.20
Nov   0.82    5.00    41.03   84.25   881.00
Dec   1.68    10.17   52.40   73.78   918.97

HOUR: 01Z         Visibility
MONTH VLIFR   LIFR    IFR     MVFR    VFR
Jan   6.43    22.72   60.88   102.60  876.63

                            ...
```

To save image to a file:

- Select the **Save Image** item.
- Type the file name in the popup dialog. To specify file format, use one of the extensions: .ps, .jpg, .png. Press **Save**.

Note

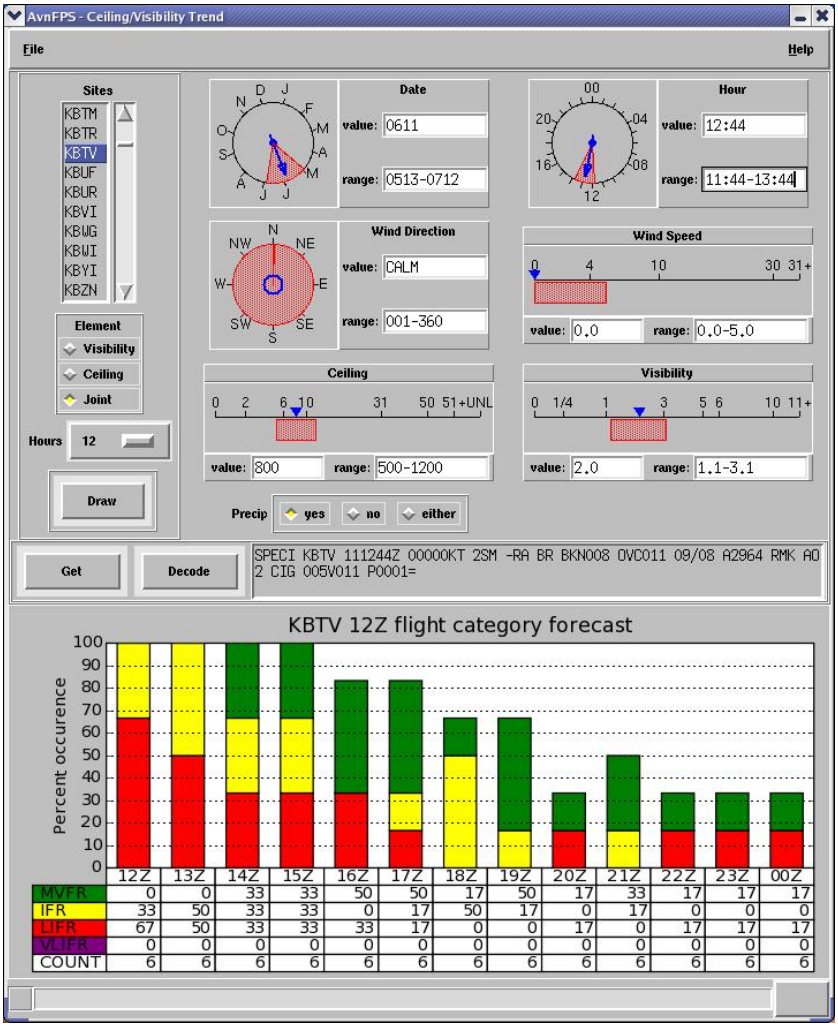
Image quality is noteably better if preserved in its native postscript format. Reformatting to either PNG or JPG format degrades the quality of the image.

To print image:

- Select the **Print Image** item.
- If needed, type in printer destination:
  `lpr -P` *`printer`*
  If the -P option is not specified, the image will be printed on your default printer.
- Press **OK**

6.5. Ceiling and Visibility Trend

The Ceiling and Visibility Trend GUI displays forecast of visibility and ceiling given initial conditions. The display is in a form of a histogram.



Visibility Trend

To display ceiling/visibility trend:

- Use the **Sites** list for the site of interest.
- Use the **Date**, **Hour**, **Wind Direction**, **Wind Speed**, **Ceiling**, **Visibility** and **Precip** widgets to set initial conditions. Each of the widgets contains a blue arror indicating current value and a red strip/sector which is used to set matching criteria for searching similar events in climatological database. Left button moves value or an edge of range. Middle button is used to move both value and range. In the "Wind Direction" widget use right button to toggle between wind arrow and a circle representing calm and variable wind.

  You can start with the most recent METAR. Select the **Get** button to retrieve the observation from the text database. The **Decode** button initializes the selection widgets.
- In the **Element** frame, select the weather element or, to display flight category, **Joint** toggle.

- Press the **Draw** button to draw the histogram.
- By default, a 6-hour forecast is displayed. You can change number of hours to 3, 6, or 12 from the **Hours** option menu.

The displayed data can be saved to a file as text or graphic and can be printed. Those features are accessible through the Menu Bar. To save image to a file:

- Select the **Save Image** item.
- Type the file name in the popup dialog. To specify file format, use one of the extensions: .ps, .jpg, .png. Press **Save**.

Note

Image quality is noteably better if preserved in its native postscript format. Reformatting to either PNG or JPG format degrades the quality of the image.

To print image:

- Select the **Print Image** item.
- If needed, type in printer destination:
  `lpr -P` *`printer`*
  If the -P option is not specified, the image will be printed on your default printer.
- Press **OK**

Appendix A. Editor Key Bindings

The editor is a standard Tk text window. Tk implements its own set of editing methods (called key bindings). The bindings below are quoted from [Welch] [131], pp 385-387. The first 3 items on the list are specific to AvnFPS.

Table A.1. Key Bindings

Ctrl-u	Undo changes.
Ctrl-r	Redo changes.
Insert	Toggles insert/overwrite mode.
Any-Key	Insert normal printing characters.
Button1	Sets the insert point, clear the selection, set focus.
Ctrl-Button1	Set the insert point without affecting the selection.
Button1-Motion	Sweep out a selection from the insert point.
Double-Button1	Select the word under the mouse.
Triple-Button1	Select the line under the mouse.
Shift-Button1	Adjust the end of selection closest to the mouse.
Shift-Button1-Motion	Continue to adjust the selection.
Button2	Paste the selection, or set the scrolling anchor.
Button2-Motion	Scroll the window.
Left or **Ctrl-b**	Move the cursor left one character. Clear selection.
Shift-Left	Move the cursor and extend the selection.
Ctrl-Left	Move the cursor by words. Clear the selection.
Ctrl-Shift-Left	Move the cursor by words. Extend the selection.
Right or **Ctrl-f**	Right bindings are analogous to Left bindings.
Alt-b or **Alt**	Same as **Ctrl-Left**, **Ctrl-Right**.
Up or **Ctrl-p**	Move the cursor up one line. Clear the selection.
Ctrl-Up	Move the cursor by paragraph which are group of lines separated by a blank line.
Ctrl-Shift-Up	Move the cursor by paragraph. Extend selection.
Down or **Ctrl-n**	All Down bindings are analogous to Up bindings.
PgUp, **PgDn**	Move the cursor by one screen. Clear the selection.
Shift-PgUp, **Shift-PgDn**	Move the cursor by one screen. Extend the selection.
Home or **Ctrl-a**	Move the cursor to line start. Clear the selection.
Shift-Home	Move the cursor to line start. Extend the selection.
End or **Ctrl-e**	Move the cursor to line end. Clear the selection.
Shift-End	Move the cursor to line end. Extend the selection.
Ctrl-Home	Move the cursor to the beginning of text. Clear the selection.
Ctrl-End	Move the cursor to the beginning of text. Extend the selection.
Ctrl-/	Select everything in the text widget.
Ctrl-	Clear the selection.
Delete	Delete the selection, if any. Otherwise delete the character to the right of the cursor.

Backspace or **Ctrl-h**	Delete the selection, if any. Otherwise delete the character to the left of the cursor.
Ctrl-d	Delete character to the right of the cursor.
Alt-d	Delete word to the right of the cursor.
Ctrl-k	Delete from cursor to the end of the line. If you are at the end of the line, delete the newline character.
Ctrl-o	Insert a newline but do not advance the cursor.
Alt-Delete	Delete the word to the left of the cursor.
Ctrl-t	Transpose the characters on either side of the cursor.

Appendix B. Monitoring Rules

Each of the monitoring modules has its own set of available rules. All rules will have the following characteristics: severity, type, unique toggle, and, optionally, message.

- Severity is the severity color which will be associated with this rule.
- Type is the name of the weather element which is monitored by this rule. With each type there is an associated label in the monitoring GUI. This is set in the configuration file `etc/gui.cfg` [13]. If a rule is used, its type must be listed on the items line.
- Unique is a toggle which determines whether the TAF Monitor will show two or more alerts of this type or limit its display to the most severe of the alerts. This feature is useful for rules that key on category differences. If one rule describes a difference of three categories as red while another rule describes a difference of two categories as yellow, situations that satisfy the red rule will always satisfy the yellow rule as well. If the Unique toggle is off, both rules will display in the TAF Monitor. If the Unique toggle is on, only the red rule will display. Message is a brief message that will be displayed in the TAF Monitor when the monitoring rule is triggered.
- Message, if set, is the text displayed in the popup window, when the rule evaluates to "True". If message is an empty string, it will be generated by the software.

This following tables describe available set of rules for each module.

Table B.1. METAR Monitoring Rules

Method:	AirportOpsThresh
Type: Unique: Args: Default Msg: Description:	cat True *VSBY*,*CIG*,*REMARKS* [METAR|TAF] LT airport operations criteria (HHH|xSM): [TAF/METAR info] Alert when either TAF or METAR cig/vis falls to or below an airport's operations criteria and the observation and forecast disagree. Variability of ceiling, visibility and sky condition may be considered if remarks fields indicates this.
Method:	CAC_AirportOpsThresh
Type: Unique: Args: Default Msg: Description:	cat True *VSBY*,*CIG*,*REMARKS* None. Messages from this rule are system generated. Alert when either TAF or METAR cig/vis falls to or below an airport's operations criteria and the observation and forecast disagree. Variability of ceiling, visibility and sky condition may be considered if remarks fields indicates this. This rule satisfies CAC requirements.
Method:	CAC_FltCatDelta
Type: Unique: Args: Default Msg: Description:	cat True *REMARKS* None. Messages from this rule are system generated Compares current forecast flight category with observed conditions. Severity of the alerts are

	automated. Variability of ceiling, visibility and sky condition may be considered if remarks fields indicates this. This rule satisfies CAC requirements.
Method:	CAC_VsbyMetarThresh
Type: Unique: Args: Default Msg:	cat False *VSBY1*,*VSBY2*,*REMARKS* None. Messages from this rule are system generated
Description:	This rule satisfies CAC requirements.
Method:	CAC_VsbyTafThresh
Type: Unique: Args: Default Msg:	cat False *VSBY1*,*VSBY2*,*REMARKS* None. Messages from this rule are system generated
Description:	This rule satisfies CAC requirements.
Method:	CAC_WxTafDelta
Type: Unique: Args: Default Msg:	wx False *WX* (a , separated list of weather elements) Weather (any on the *WX* list) forecasted but does not occurs in METAR
Description:	This rule satisfies CAC requirements.
Method:	CigCatDelta
Type: Unique: Args: Default Msg:	sky True *NCAT*,*REMARKS* TAF and METAR differ by *NCAT* categories
Method:	CigMetarThresh
Type: Unique: Args: Default Msg:	sky False *CIG1*,*CIG2*,*REMARKS* METAR ceiling <= *CIG1* and TAF ceiling > *CIG2*.
Description:	Compares current forecasted ceiling value with observed conditions. Variability of ceiling and sky condition may be considered if remarks field indicates this.
Method:	CigTafThresh
Type: Unique: Args: Default Msg:	sky False *CIG1*,*CIG2*,*REMARKS* TAF ceiling <= *CIG1* and METAR ceiling > *CIG2*.
Description:	Compares current forecasted ceiling value with observed conditions. Variability of ceiling and sky condition may be considered if remarks field indicates this.
Method:	DDDelta
Type: Unique: Args:	wind True *DD*,*FF1*

Default Msg:	Wind directions differ by dd degrees, [METAR|TAF] winds >= *FF1* kts.
Method:	FFDelta
Type: Unique: Args: Default Msg:	wind True *FF*, *FF1* Wind speeds differ by *FF* kts, [METAR|TAF] wind >= *FF1*.
Method:	FltCatDelta
Type: Unique: Args: Default Msg: Description:	cat True *REMARKS*,*STRICT* [Cig|Vis|Flight] category differ: [TAF/METAR info] Compares current forecast flight category with observed conditions. Severity of the alerts are automated. Variability of ceiling and sky condition may be considered if remarks field indicates this. The strict flag means that to avoid alerts, the forecasted and observed categories must match exactly.
Method:	FuelAlternate
Type: Unique: Args: Default Msg: Description:	cat True *VSBY*,*CIG*,*REMARKS* [METAR|TAF] LT xSM/HHH requires additional fuel and alternate airport: [TAF/METAR info] An alert is given when cig/vis conditions are unforecasted or unobserved. Variability of ceiling and sky condition may be considered if remarks field indicates this.
Method:	LFFMetar
Type: Unique: Args: Default Msg: Note:	wind False *RUNWAY*, *FF* [Headwind|Tailwind] *XX* KT on runway *RUNWAY* >= *FF*. *RUNWAY* is an array *index* for the runway attribute in the airport's Site Info File [18], not the actual runway heading itself. *FF* < 0 indicates headwind; *FF* > 0 indicates tailwind.
Method:	VsbyCatDelta
Type: Unique: Args: Default Msg:	vsby True *NCAT*,*REMARKS* TAF and METAR differ by *NCAT* categories
Method:	VsbyMetarThresh
Type: Unique: Args: Default Msg:	vsby False *VSBY1*,*VSBY2*,*REMARKS* METAR visibility <= *VSBY1* and TAF visibility > *VSBY2*.
Method:	VsbyTafThresh
Type: Unique: Args:	vsby False *VSBY1*,*VSBY2*,*REMARKS*

Default Msg:	TAF visibility <= *VSBY1* and METAR visibility > *VSBY2*.
Method:	WxMetar
Type: Unique: Args: Default Msg:	wx False *WX* (a , separated list of weather elements) Weather (any on the *WX* list) occurred in METAR
Method:	WxMetarDelta
Type: Unique: Args: Default Msg:	wx False *WX* (a , separated list of weather elements) Weather (any on the *WX* list) occurs in METAR but not forecasted
Method:	WxTafDelta
Type: Unique: Args: Default Msg:	wx False *WX* (a , separated list of weather elements) Weather (any on the *WX* list) forecasted but does not occurs in METAR
Method:	WxVsbyDelta
Type: Unique: Args: Default Msg:	wx False *WX* (a , separated list of weather elements), *VSBY*,*REMARKS* Weather (any on the *WX* list) occurred in METAR but not forecasted. Visibility <= *VSBY*
Method:	XFFMetar
Type: Unique: Args: Default Msg: Note:	wind False *RUNWAY*, *FF* Crosswind X kts on runway *RUNWAY* >= *FF*kts. *RUNWAY* is an array *index* for the runway attribute in the airport's Site Info File [18], not the actual runway heading itself.

Table B.2. GFE Grid Monitoring Rules

Method:	DDDelta
Type: Unique: Args: Default Msg:	wind True *DD*, *FF1* TAF and GRIDs; wind directions differ by *DD* with [TAF|METAR] wind exceeds *FF1* kt.
Method:	FFDelta
Type: Unique: Args: Default Msg:	wind True *FF*, *FF1* [TAF|METAR] wind speeds differ by *FF*, [METAR|TAF] wind speed exceeds *FF1*.
Method:	CigCatDelta

Type: Unique: Args: Default Msg:	sky True *NCAT* TAF and METAR ceilings differ by *NCAT* categories
Method:	SkyMismatch
Type: Unique: Args: Default Msg:	sky True *NCAT* TAF and GRIDs sky covers differ by *NCAT* categories.
Method:	VsbyCatDelta
Type: Unique: Args: Default Msg:	vsby True *NCAT* TAF and METAR visibilities differ by *NCAT* categories
Method:	WxTafDelta
Type: Unique: Args: Default Msg:	wx False *WX* (a , separated list of weather elements) Weather (any on the *WX* list) occurs in TAF and not in GRIDs
Method:	WxGridsDelta
Type: Unique: Args: Default Msg:	wx False *WX* (a , separated list of weather elements) Weather (any on the *WX* list) occurs in GRIDs and not in TAF

Table B.3. Lightning monitoring rules

Method:	TsObsDelta
Type: Unique: Args: Description:	wx True *NUM* Number of reported lightning strikes within last 10 min >= *NUM*

Table B.4. Lightning probability forecast monitoring rules

Method:	TsInTaf
Type: Unique: Args: Description:	wx True *PROB* Thunder in TAF but guidance probability < *NUM*
Method:	TsNotInTaf
Type:	wx

Unique:	True
Args:	*PROB*
Description:	Thunder in not TAF but guidance probability >= *PROB*

Table B.5. CCFP monitoring rules

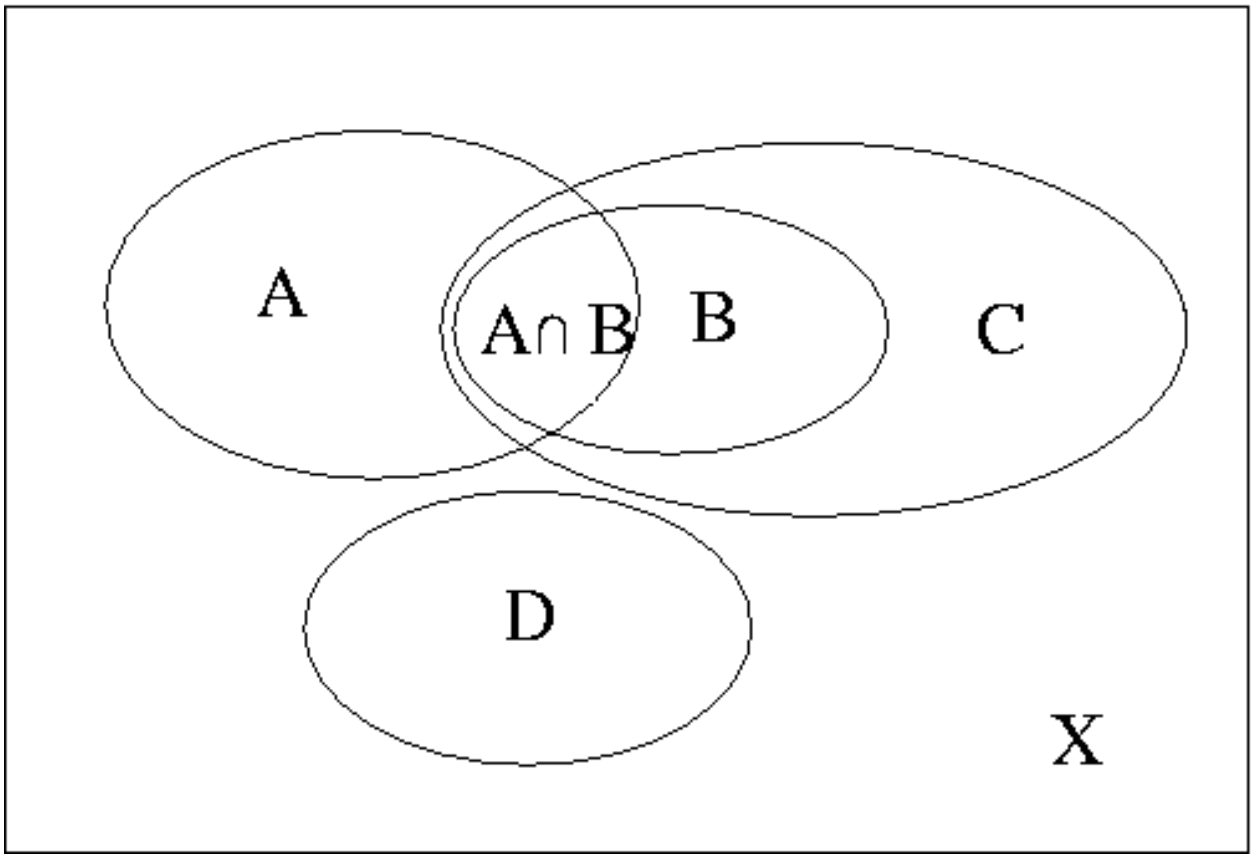
Method:	CBNotInTaf
Type:	wx
Unique:	True
Args:	*CONF*,*CVRG*
Description:	CCFP exceedes confidence and coverage thresholds with no CB in TAF.
Method:	TsNotInTaf
Type:	wx
Unique:	True
Args:	*CONF*,*CVRG*
Description:	CCFP exceedes confidence and coverage thresholds with no TS in TAF.

Table B.6. Low Level Wind Shear Monitoring Rules

Method:	WSinRadar
Type:	wx
Unique:	True
Args:	*VALUE* (1/s)
Description:	Derived wind shear value >= *VALUE*

Appendix C. Climate Quality Control

The algorithm used to check climatological consistency of a forecast is based on conditional probability estimates. The best way to illustrate the idea is to look at Venn's diagrams.



Venn diagram

An *event* is a subset of an *outcome space*, that is a set of all possible outcomes of an experiment. An event determined by random variable x is of the form $A = \{x \in A\}$. When the random variable is observed, that determines whether or not A occurs: if the value of x happens to be in A, A occurs, if not, A does not occur. Probability of an event (i.e. set) is its area, normalized such that the area of the outcome space X is one. Given two events A and B, the conditional probability of A, given B is defined by the formula

$$P(A \mid B) = P(A \cap B) / P(B)$$

Events A and B are said to be *independent* if

$$P(A \cap B) = P(A) * P(B) \Leftrightarrow P(A \mid B) = P(A)$$

Disjoint events, such as A and D on the above diagram, are *not* independent, they are exclusive: if D has happened, than A could not have happened. In this case

$$P(A \cap D) = 0 \Rightarrow P(A \mid D) = 0.$$

The other extreme case, where one set contains the other, illustrates dependent events: if B has happened, so must have C.

$$P(C \cap B) = P(B) \Rightarrow P(C \mid B) = 1.$$

These relations form the basis of our algorithm: we will claim that the combination of events is unlikely if

$P(A \cap B) < \alpha_0 * P(A) * P(B)$

or equvalently,

(*) $P(A \mid B) < \alpha_0 * P(A)$

where α_0 is a fixed arbitary value such that $0 < \alpha_0 < 1$.

The algorithm considers seven weather elements: ceiling, visibility, wind direction and speed, obstruction to vision, precipitation type and intensity. Obstruction to vision can be one of HZ, FG/BR, BLSN/BLSA or None. Precipitation type is one of SN, DZ, RA. Mixed precipitation is coerced to one of these types. Precipitation intensity is None, light, moderate or severe. Continuous elements are categorized: given a sequence of thresholds $0 < \tau_1 < \ldots < \tau_n$, the category k is a set

$A_k = \{v: \tau_k <= v < \tau_{k+1}\}$, k = 0, ..., n+1,

with $\tau_0 = 0$ and $\tau_{n+1} = \infty$. Wind categories are a special case: instead of adding 0 and ∞ wind direction categories have to be "wrapped" modulo 360 degrees. An additional category is added for calm or light (that is, category 0) winds.

These categories form our outcome space, that is any weather instance can be qualified by a combination of the categories (i.e. events). Denote

$\{FF_k\}_{k=0,\ldots n1}$ - set of all wind speed events
$\{DD_k\}_{k=0,\ldots n2}$ - set of all wind direction events
$\{VSBY_k\}_{k=0,\ldots n3}$ - set of all visibility events
$\{CIG_k\}_{k=0,\ldots n4}$ - set of all ceiling events
$\{OBV_k\}_{k=0,\ldots n5}$ - set of all obstruction to vision events
$\{PTYP_k\}_{k=0,\ldots n6}$ - set of all precipitation type events
$\{PINT_k\}_{k=0,\ldots n7}$ - set of all precipitation intensity events
Let A_i be one of the events on the above list. Define $B_i = A_{k1} \cap \ldots \cap A_{k6}$, where k_j != i. Then $A_i \cap B_i = A_1 \cap \ldots \cap A_7$ By substituting these formulas into (*) we come up with 7 conditional probabilities.

(**) $P(A_i \mid B_i) < \alpha_0 * P(A_i)$, i = 1, ..., 7.

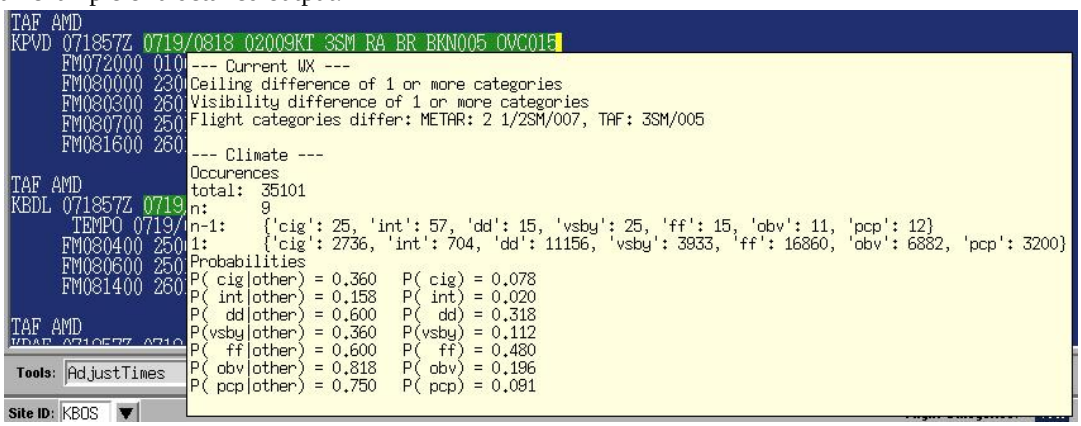
If the above inequality does not hold for some i we flag the corresponding weather element as suspicious.

The algorithm works as follows:

- Categorize weather in a TAF period.
- Evaluate minimal value α_{min} such that (**) is true for all i.
- If $\alpha_{min} \geq \alpha_0$, the forecast is plausible, otherwise continue.
- If $\alpha_{min} = 0$, none of the historical data matches forecasted weather. The software highlights the forecast in orange. If $0 < \alpha_{min} < \alpha_0$, the combination of weather elements is not likely and the forecast highlighted in green.
- An attempt is made to suggest more plausible weather: For each of the following elements: wind speed, direction, ceiling, visibility and precipitation intensity, a neighboring category is considered. The probabilities are calculated as above and a corresponding α_{min} value is calculated. If it is greater or equal to α_0, the weather combination is saved on a list.
- The list is sorted with respect to α and made available to the forecaster in the TAF Editor window.

The category thresholds and the value of α_0 are configurable, see System Administration, the section called “Climate QC Definition File” [20].

Here is an example of a detailed output:



Detailed Output

Appendix D. Low Level Wind Shear Monitoring

Low-Level Wind Shear (LLWS) monitoring has been implemented since AvnFPS 3.0. The WSR88D VWP tabular information, wind profiler data, and Aircraft Communications Addressing and Reporting System (ACARS) data along with surface observations at TAF sites allow vertical wind shear to be calculated. With appropriate thresholds, AvnFPS can alert the forecaster to the presence or the possibility of LLWS in the vicinity of their airports. As defined in [NWSI 10-813] [131], Section 1.2.8, LLWS is wind shear of 10 kts or more per 100ft in a layer more than 200 feet thick. Only winds within the lowest 2000 feet are used for LLWS calculations. LLWS monitoring alerts appear on AvnFPS Monitoring GUI, in a manner similar to the lightning alerts.

Important

Remember that vertical wind shear is not a scalar quantity, but a vector. Using just the difference in magnitude, i.e. speed shear, will, in the majority of cases, underestimate the amount of shear present. Direction of the horizontal winds must be considered as well.

On a benign weather day, wind shear values typically are less than 0.08 s^{-1}. Wind shear meeting official criteria is 0.169 s^{-1}. The LLWS monitoring algorithm uses two threshold values which provides three wind shear regimes. These thresholds separates the wind shear spectrum into normal (green) and caution (yellow) and warning (red) regions. The placement of the thresholds is set via the AvnFPS configuration GUI and are customizable for each TAF site, see Section 4.1: “Editing Monitoring Rules” (page 32).

As you move your mouse over the LLWS alert light, the current TAF is displayed along with the source identifier producing the greatest shear and a "best-guess" (based on available data) wind shear group, suitable for placing into the TAF. The LLWS alert light will only change if elevated wind shears are detected and there is no WS group in your TAF forecast valid at that time. If there is a WS group in the TAF, the LLWS Alert light will *not* change, i.e. it will stay green, *regardless of the amount of wind shear detected*.

It is the WFO's aviation focal point's judgement to decide whether to use a radar's VWP product or profiler's data for a particular TAF site. Physical promixity and elevation of the data sources--radars and profilers--need to be considered as to whether LLWS monitoring is practical (or effective) for a particular TAF site. Placement of a radar on a mountaintop would be a disqualifying factor for use in LLWS monitoring since winds from such a location are likely to be unrepresentative of conditions over an airport that is situated at lower elevation. Keep in mind that wind shear impacting aircraft operations is limited to altitudes of 2000 ft AGL and lower. For a given TAF site, one can configure multiple upper-air sources to be used with the airport's surface observations. In such cases, the alert level on the AvnFPS Monitoring GUI is based on the maximum shear found.

To begin LLWS monitoring for your TAF sites, see Section 4.2: “Editing TAF Site Information” (page 35).

In the event that the LLWS monitoring icon goes yellow or red, the forecaster should consult the VWP product of the profiler, radar, or ACARS that triggered the alert.

Important

If LLWS alert is based on radar data, the forecaster should examine the radar's VWP *text* product, not the graphic that's displayed in D-2D. The radar VWP text product shows greater vertical resolution in the lowest 2000 feet than what is displayed in the graphic.

With these data, the forecaster can judge the validity of the wind data and susequently the LLWS alert. If there is corroborative evidence from other sources that strong, non-convective wind shears are occurring, then amending the TAF to put in a WS group should be considered.

Appendix E. Ceiling and Visibility Grid Monitoring

One of the new features implemented with AvnFPS 3.2, is the ability to monitor IFPS ceiling and visibility grids in the AvnWatch GUI. This is an experiment pioneered by the Charleston, WV (RLX) WFO with a few other eastern region WFOs participating. The acquisition of such grids into GFE is described. However, the initialization, methodology or editing of such grids are not discussed here.

The required 'parms', predominate and categorical cloud heights and categories, and visibility grids, need to be added to the GFESuite ifpServer Fcst and Official databases. To do this, the GFESuite's localConfig.py file need the following entries:

```
CigHgt = ("CigHgt", SCALAR, "100ft", "Cloud Ceiling Height", 250.0, 0.0, 0, NO)
PredHgt = ("PredHgt", SCALAR, "100ft", "Predominant Cloud Height", 250.0, 0.0, 0, NO)
CigHgtCat = ("CigHgtCat", SCALAR, "index", "Cloud Ceiling Height Category", 6.0, 0.0, 0, NO)
PredHgtCat = ("PredHgtCat", SCALAR, "index", "Predominant Cloud Height Category", 6.0, 0.0, 0, NO)
Vsby = ("Vsby", SCALAR, 'mi', 'Visibility', 10.0, 0.0, 2, NO)
```

And the following tuple added to the list 'parms':

```
([CigHgt, PredHgt, CigHgtCat, PredHgtCat, Vsby], TC1)
```

Once updated, the ifpServer will have to be stopped and restarted for these changes to take effect.

A GFESuite text formatter, `/awips/adapt/avnfps/OB9.2/py/IFPS2AvnFPS.py`, will need to be modified to query the new ifpServer parms and place them into the AvnFPS database. The octalthorpes, #, in the first column from lines 297 to 317 and lines 333 to 337 need to be removed. Save your changes.

To monitor these grids in AvnWatch GUI, new monitoring rules for grids need to be added. Using the the AvnFPS configuration tool 'Monitoring Critera' GUI, select the **grids** tab. A list of current monitoring rules for grids appear in the upper-half of the editor. In the available methods column, select the 'CigCatDelta' rule and then press **Add** button to monitor ceiling categories between your TAFs and the GFE grids. To monitor GFE visibility grids with your TAFs, select the 'VsbyCatDelta' monitoring rule and press the **Add** button. Press the **Save** to save your changes and restart the AvnWatch GUI.

Appendix F. Writing AvnFPS TAF Edit Tools

The AvnFPS TAF Editor has several tools which perform convenient, commonly performed tasks implemented as modules in the `/toolpy` directory, separate from the `/py` code base. This was done to allow innovative individuals at the WFOs to create their own tool(s) to satisfy local needs. This appendix will describe the basic requirements for writing a tool to be used in the TAF Editor and then followed-up with a code “walkthrough” of the **AdjustTimes** Tool.

For a new tool to be invoked from the TAF Editor, the module, residing in the `/toolpy` directory, must have a function named “updateTafs”. This function, to be called from the TAF Editor, has fixed inputs which the caller will supply. In the table below, the arguments are described along with actual argument values as an example of what to expect when writing a new one.

```
def updateTafs(master, bbb, fcsts)
```

Table F.1. Argument List Description

Name	*Input/Output*	*Description*
master	Input	widget object to create a GUI for tool, if needed, as a child of the TAF Editor
bbb	Input	text string object denoting whether the output TAF(s) is formatted as either initial issuance, amended, corrected or delayed transmission product(s)
fcsts	Input/Output	dictionary object with IDs and forecast text serving as keys and values, respectively

The dictionary labelled “fcsts” is returned to the caller with changed forecast(s) and displayed in the TAF Editor text window.

What follows is the actual program listing of the tool called **AdjustTimes** with explanations. For any new tools, it is good practice and consistent with AvnFPS behavior to log errors and report messages back to the TAF Editor as done here. Use of existing functions in AvnLib.py, AvnParser.py and other baseline modules in locally-defined tools is strongly encouraged. For any tool to be used in the TAF Editor, the module should be placed in the `/toolpy` directory. Once there, it will be available as an item in the pull-down list in the editor.

```
import logging
import Avn, AvnLib, Busy, TafDecoder, AvnParser ❶

_Logger = logging.getLogger(__name__) ❷

def updateTafs(master, bbb, fcsts):
badidents = []
decoder = TafDecoder.Decoder() ❸

for key in fcsts:
    kind, ident = key
```

```
❹
        try:
            tafDuration=int(AvnParser.getTafSiteCfg(ident)['thresholds']['tafduration'])  ❺

        except:
            tafDuration=24

        taf = decoder(fcsts[key], bbb)                                                  ❻

        if not 'group' in taf or not taf['group']:
            _Logger.error('NIL TAF for %s', ident)
            continue

        AvnLib.adjustTimes(bbb, taf)                                                    ❼

        evtime=taf['vtime']['str'][5:]
        lines = AvnLib.makeTafFromPeriods(ident, bbb, taf['group'],
                                          tafDuration=tafDuration,
                                          evtime=evtime)                                ❽

        if 'amd' in taf:                                                                ❾

            m = _AmdPat.search(taf['amd']['str'])
            if m:
                vtime = taf['vtime']['from']
                tms = list(time.gmtime(vtime))
                #
                # Get new valid time made by adjustTimes routine
                vpttrn = _AmdPat.search(' '.join(lines[:2]))
                if vpttrn:
                    s = vpttrn.group(3)
                    tms[2:4] = (int(s[:2]), int(s[2:4]))
                    TafDecoder.fix_date(tms)
                    vtime = time.mktime(tms)
                    #
                    # If a TIL or AFT expression
                    if m.group(1):

                        s = m.group(2)
                        tms[2:5] = (int(s[:2]), int(s[2:4]), int(s[4:6]))
                        TafDecoder.fix_date(tms)
                        amdtime = time.mktime(tms)

                        if m.group(1) == 'AFT' and vtime >= amdtime :
                            lines.append(taf['amd']['str'][:m.start(1)])
                        elif m.group(1) == 'TIL' and vtime < amdtime:
                            lines.append(taf['amd']['str'])
                    #
                    # Range
                    elif m.group(3):

                        s = m.group(3)
                        tms[2:4] = (int(s[5:7]), int(s[7:]))
                        TafDecoder.fix_date(tms)
                        amdtime = time.mktime(tms)
                        if vtime < amdtime:
                            tms[2:4] = (int(s[:2]), int(s[2:4]))
                            TafDecoder.fix_date(tms)
                            amdtime = time.mktime(tms)
                            if vtime >= amdtime:
                                lines.append('%sTIL %s00' % (taf['amd']['str'][:m.start(3)],s[5:]))
                            else:
                                lines.append(taf['amd']['str'])
                else:
                    lines.append(taf['amd']['str'])
            else:
                    lines.append(taf['amd']['str'])

        fcsts[key] = '\n'.join(AvnLib.indentTaf(lines)+['''])                           ❶

    except Avn.AvnError, e:
        badidents.append(ident)

if badidents:                                                                           ❶❶

    Busy.showwarning('Could not update times for %s' % ' '.join(badidents),
        master)

return fcsts                                                                            ❶❷
```

❶ Include needed AvnFPS (or Python) modules to accomplish task
❷ Initialize logging object so that any error/warning/info message can be recorded in a log file.
❸ Create an instance of the TAF decoder
❹ For each TAF found in the text editor window....

5. ...get the nominal duration of the forecast
6. ...decode the current forecast
7. ...call function to remove obsolete forecast groups from the product
8. ...rebuild the TAF text
9. ...check references to time in the AMD clause
10. ...update the fcsts dictionary with the new product
11. ...record which TAFs had problems and display them to the user
12. Return the updated TAFs to caller

The **AdjustTimes** tool is one of several in the directory `/toolpy`. There are others and are described below.

Table F.2. Description of the Tools in `/toolpy` directory

Name	Description and Purpose
AdjustTimes	Remove obsolete forecast groups from the TAFs.
CopyForecasts	Copy one TAF to many.
UseMetarForPrevailing	Replace the TAF's initial forecast group with values to be consistent with latest observation.
WestFlow	WFO RLX specific tool to use KPKB and KHTS TAFs as sources to initialize other RLX forecasts by adjusting forecast groups' times and cloud heights.

Bibliography

[CCFP] *Collaborative Convective Forecast Product*. Document [http://aviationweather.gov/products/ccfp]

[Gamin] Daniel Veillard, *Gamin 0.1.10*. November 24, 2008. Web Site [http://www.gnome.org/~veillard/gamin/]

[Java] *Java* Sun Microsystems. Web Site [http://www.java.com/en]

[LAMP] Judy Ghirardelli. National Weather Service, Meteorological Development Laboratory. "An overview of the Redeveloped Localized Aviation MOS Program (LAMP) for Short-Range Forecasting". Preprints, 21st Conference on Weather Analysis and Forecasting. Washington, D.C. American Meteorological Society, 2005. Paper [http://www.nws.noaa.gov/mdl/pubs/Documents/Papers/LAMP_overview_paper_AMS2005_final.pdf]

[Charba, et al.] Jerome Charba. Frederick Samplatsky. National Weather Service, Meteorological Development Laboratory. "Operational 2-h thunderstorm guidance forecasts to 24 hours on a 20-km grid", Preprints, 23rd Conference on Weather Analysis and Forecasting. Omaha, Nebraska. American Meteorological Society, 2009. Paper [http://www.nws.noaa.gov/mdl/pubs/Documents/Papers/Charba2009Operational2-h.pdf]

[Glahn, et. al.] Harry R. Glahn, David P. Ruth. National Weather Service, Meteorological Development Laboratory "The New Digital Forecast Database of the National Weather Service" Bulletin of the American Meteorological Society: Vol. 84, No. 2, pp. 195-201, 2003. Paper [http://www.nws.noaa.gov/mdl/pubs/Documents/Papers/Glahn2003TheNew.pdf]

[NWSI 10-813] "National Weather Service Instructions 10-813" Operations and Services, Aviation Weather Services. December 2008. Document [http://nws.noaa.gov/directives/sym/pd01008013curr.pdf]

[Pyro] Irmen de Jong. *Python Remote Objects version 3.5*. Feb 2005. Web Site [http://pyro.sourceforge.net]

[Python] *Python 2.4.1*. March 2005, Python Software Foundation. Web Site [http://www.python.org]

[Smirnova, et. al.] Tanya Smirnova, Stan Benjamin, John Brown, "Case study verification of RUC/MAPS fog and visibility forecasts". Preprints, Ninth Conference on Aviation, Range and Aerospace Meteorology. Orlando, FL. American Meteorological Society, 2000.

[TPG] Christophe Delord, *Toy Parser Generator 3.1.0*. March 2006. Web Site [http://christophe.delord.free.fr/tpg/index.html]

[Oberfield, et. al.] Mark Oberfield, Matthew Peroutka, Cynthia Abelman. National Weather Service, Meteorological Development Laboratory, Office of Science and Technology, Aviation Services. "Using probablistic guidance and an update technique to generate terminal aerodrome forecasts." Preprints, 13th Conference on Aviation, Range and Aerospace Meteorology. New Orleans, LA., American Meteorological Society, January 2008. Paper [http://www.nws.noaa.gov/mdl/pubs/Documents/Papers/Oberfield2008UsingProbabilistic.pdf]

[Welch] Brent B. Welch. *Practical Programming in Tcl and Tk*. Second Edition. Prentice Hall PTR. 1997. ISBN: 0-13-616830-2