# DCS Internet Toolkit User’s Guide

## Table of Contents

1. **INTRODUCTION** ................................................................................................................... 1

2. **TOOLKIT INSTALLATION** ..................................................................................................... 2
   - 2.1 Installing the Java Runtime Environment ......................................................................... 2
   - 2.1.1 Installing the Java SDK on Windows .............................................................................. 2
   - 2.1.2 Installing the Java SDK on Linux .................................................................................. 3
   - 2.2 Backup Your Current Installation ..................................................................................... 4
   - 2.3 Install the Toolkit ............................................................................................................... 4
   - 2.4 Setting Your Runtime Environment .................................................................................... 6
   - 2.4.1 Setting the PATH Variable on Windows 2000 and XP .................................................... 6
   - 2.4.2 Setting the PATH Variable on Linux and Unix ............................................................... 6
   - 2.5 Activating the Toolkit License ............................................................................................ 7
   - 2.6 Set Up the Toolkit for a SQL Database ............................................................................... 7
     - 2.6.1 Install PostgreSQL ......................................................................................................... 7
     - 2.6.2 Configure the PostgreSQL Server .................................................................................. 9
     - 2.6.3 Create DECODES Administrative Database User .......................................................... 9
     - 2.6.4 Configure DECODES to use SQL .................................................................................. 10
     - 2.6.5 Create the DECODES Database Schema ..................................................................... 10

3. **RUNNING THE TOOLKIT GRAPHICAL USER INTERFACE (GUI)** ......................................... 12

4. **DCS TOOLKIT SETUP** .......................................................................................................... 13
   - 4.1 Software Updates ................................................................................................................ 14
   - 4.2 License Activation ................................................................................................................ 15
   - 4.3 DECODES Properties ......................................................................................................... 16

5. **RETRIEVAL PROCESSES** ..................................................................................................... 17
   - 5.1 What is a Retrieval Process? ................................................................................................ 17
   - 5.2 The Retrieval Process Scheduler ....................................................................................... 18
   - 5.3 The Retrieval Process Monitor and Control Screen ............................................................. 18
   - 5.4 Creating or Editing Retrieval Process .................................................................................. 19
   - 5.5 Starting & Stopping Retrieval Processes ............................................................................. 23
   - 5.6 Monitor a Running Retrieval Process .................................................................................. 23

6. **THE DECODES DATABASE** .................................................................................................. 24
   - 6.1 Data Server Records .......................................................................................................... 24
   - 6.2 Running DECODES Programs from the Command Line ....................................................... 26

7. **MAINTAINING NETWORK LISTS** .......................................................................................... 27

8. **BROWSING FOR DCP MESSAGES** .................................................................................... 28
   - 8.1 The Message Output Screen .............................................................................................. 29

9. **SEARCH CRITERIA** ................................................................................................................. 30
   - 9.1 Search Criteria Graphical Editor .......................................................................................... 30
     - 9.1.1 Specifying Time Range ................................................................................................... 31
     - 9.1.2 Specifying Platform(s) by Name, Address, or List .......................................................... 32
     - 9.1.3 Specify GOES Channels, Spacecraft, or Baud Rate ....................................................... 32
     - 9.1.4 Qualifiers and Other ....................................................................................................... 33
   - 9.2 Search Criteria File Format .................................................................................................. 34
     - 9.2.1 DRS verses DAPS Time .................................................................................................. 35
     - 9.2.2 Allowable Time Formats for a Search Criteria File ......................................................... 36

### Allowable Time Formats for a Search Criteria File

- **DRS verses DAPS Time**
- **Specify GOES Channels, Spacecraft, or Baud Rate**
- **Qualifiers and Other**

### Setting Your Runtime Environment

- **Setting the PATH Variable on Windows 2000 and XP**
- **Setting the PATH Variable on Linux and Unix**

### Activating the Toolkit License

- **Install PostgreSQL**
- **Configure the PostgreSQL Server**
- **Create DECODES Administrative Database User**
- **Configure DECODES to use SQL**
- **Create the DECODES Database Schema**

### Set Up the Toolkit for a SQL Database

- **Install PostgreSQL**
- **Configure the PostgreSQL Server**
- **Create DECODES Administrative Database User**
- **Configure DECODES to use SQL**
- **Create the DECODES Database Schema**

### Creating or Editing Retrieval Process

- **What is a Retrieval Process?**
- **The Retrieval Process Scheduler**
- **The Retrieval Process Monitor and Control Screen**
- **Creating or Editing Retrieval Process**
- **Starting & Stopping Retrieval Processes**
- **Monitor a Running Retrieval Process**

### The DECODES Database

- **Data Server Records**
- **Running DECODES Programs from the Command Line**

### Maintaining Network Lists

- **The Message Output Screen**

### Browsing for DCP Messages

- **The Message Output Screen**

### Search Criteria

- **Search Criteria Graphical Editor**
- **Search Criteria File Format**
- **Allowable Time Formats for a Search Criteria File**

### DRS verses DAPS Time

- **Specify GOES Channels, Spacecraft, or Baud Rate**
- **Qualifiers and Other**
## START TOOLKIT PROGRAMS FROM THE COMMAND LINE

- **10.1 THE ‘GETDCPMESSAGES’ UTILITY**

## DCS TOOLKIT EXTENSIONS TO DECODES

### 11.1 Output Formatters
- **11.1.1 Output Format: SDML-1 (Site Data Markup Language)**
- **11.1.2 Campbell Scientific Comma-Separated-Value Format**
- **11.1.3 Alberta Environment “Alta-Loader” File Format**
- **11.1.4 Kisters Wiski ZRXP Format**
- **11.1.5 Raw Formatter**
- **11.1.6 Pi Historian Formatter**
- **11.1.7 Vaisala “METMAN” Formatter**
- **11.1.8 SFWMD XML Formatter**
- **11.1.9 Generic Comma-Separated-Value Format**

### 11.2 Data Consumers
- **11.2.1 Socket Server Consumer**
- **11.2.2 Soap Client Consumer**

### 11.3 DECODES Format Language Extensions
- **11.3.1 OTT Pseudo Binary Bulk-Encoded Messages**

### 11.4 URL Data Sources
- **11.4.1 One-Time Setup for URL Data Sources**
- **11.4.2 Simple URL Data Source**
- **11.4.3 Template URL Data Source**
- **11.4.4 Canadian Weather Service URL Downloader**
1 Introduction

The DCS Internet Toolkit is a suite of software that allows you to retrieve DCP data over the network from any of the following:

- NOAA Public DCP Data Servers in Wallops, VA
- LRGS DOMSAT Receive Stations
- LRGS GOES Direct Readout Ground Stations
- LRGS/LRIT (Low Rate Information Transfer) Receivers
- DAPS-II IXS Systems

Please see our separate spec-sheets on our Tempest family of satellite receivers for more information. You install the Internet Toolkit on client machines that need to process or display your DCP data.

The Toolkit is implemented in 100% pure Java, so it will run on any modern computing platform (Solaris, Linux, Windows, AIX, etc.) It provides a low cost method for retrieving and saving DCP data, either in periodic batches or in real-time.

The toolkit can save raw or decoded data into local files on your machine, or you can pipe the data into your own programs in real-time. You can also integrate the toolkit classes with your own Java software for a more efficient implementation. Ilex Engineering can assist in this effort.

DCS Internet Toolkit Features:

- Retrieve DCP data from Wallops, LRGS, or IXS systems in real-time, interactively, or in periodic batches
- Fully integrated with DECODES to convert raw data to time-tagged engineering units
- Save DCP data to a local file, or pipe to your own programs in real-time
- Easy to use scheduler will automatically run your retrieval processes at set times of the day.
- Receive data from a list of servers, automatically switching to a backup server in case of failures.
- Select DCP messages by combination of time range, network list, DCP name, DCP address, GOES channel, or data source.
- Maintain your network lists with GUI editor, and automatically upload them to your data servers.
- Command-line interface to run retrieval process from within your own scripts.
- Javadoc documentation to help you integrate toolkit classes with your own Java programs
- GUI to monitor real-time status of LRGS or IXS servers, and to select most reliable data source.
- One year technical support
2 Toolkit Installation

The toolkit will run on most modern computing platforms. This section explains the installation procedure in general terms, with sub-sections for specific operating systems.

2.1 Installing the Java Runtime Environment

The toolkit software is made up of Java archives and scripts for various operating systems. To run the Java code you will need to install Sun Microsystem’s Java Runtime Environment version 1.6 or higher. This is available as a free download from:

http://java.sun.com

Download the “J2SE” Standard Edition. The latest stable version at the time of this writing is 1.6 update 7. You may download the “JRE”, or if you are interested in doing Java development, you can download the “SDK” (Software Development Kit), which contains the JRE plus several development tools.

2.1.1 Installing the Java SDK on Windows

Follow these instructions to install the SDK on Windows 2000, NT or XP:

- Download the SDK release, as described above. Download the file to your desktop or a temporary directory on your hard disk.
- Double click the icon to start the installation procedure.
- Read and agree to the Sun Microsystems License.
- Choose a destination folder for the Java SDK, or accept the default shown.
- Complete the release via the dialogs.

After installation, open a DOS window and type the command:

djava -version

You should see a version message matching the release that you installed. If you see a message that ‘java’ is not recognized as an internal or external command, etc; then Java is NOT installed properly. Review the installation instructions above.
2.1.2 Installing the Java SDK on Linux

If you want to run the toolkit on a Linux machine that is already set up as an LRGS, Java should already be installed. If not, download the JDK for Linux from http://java.sun.com. Download the ‘self-extracting RPM file. It will be stored in a file named something like:

jdk-6u7-linux-i586-rpm.bin

Note -- the release numbers may be different by the time you read this. Make a note of the release you download and make substitutions to the file and directory names in these instructions.

Login as ‘root’ and move this file to the /root directory. Then run the downloaded script to unpack and install the RPM (RedHat Package Manager) file.

```
mv jdk jdk-6u7-linux-i586-rpm.bin /root

cd /root

sh jdk-6u7-linux-i586-rpm.bin

(answer the questions about licensing agreement)
```

This will result in the Java release installed in its own subdirectory under /usr/java. We recommend that you set up a symbolic link pointing to this directory called /usr/java/jdk. You can do this as follows (note, you must be root to do this). For example:

```
cd /usr/java

ln -s jdk1.6.0_07 jdk
```

Configure your Login Account for Java

You need to place the bin directory in the Java release into your PATH variable:

```
export PATH=/usr/java/jdk/bin:$PATH
```

Place this command in your .bashrc file to have it done every time you login.

Verify that the path is properly set by typing:

```
java -version
```
2.2  **Backup Your Current Installation**

If you are upgrading from a previous version of the toolkit, we strongly recommend that you take a Zip backup of the entire installation directory before continuing.

2.3  **Install the Toolkit**

The Toolkit is distributed as an IzPack self-installing Jar file called “dcstool-4-1.jar”. Download this from the Ilex Engineering web site (www.ilexeng.com).

After downloading, start the installation from the command line:

```
java -jar dcstool-install-4-4.jar
```

This will launch a series of dialogs to guide you through the install. After accepting the license agreement you are asked to choose a directory in which to install the toolkit:

For Windows systems, we recommend C:/DCSTOOL, which is the default. You can install it in a different directory, but please avoid directories with spaces in the name.

On Unix or Linux systems, we recommend a sub-directory called DCSTOOL under your home directory.

The software will warn you if a previous version of the software resides at this location. Again, if you are upgrading from a previous toolkit release, we strongly advise you to take a zip backup of the entire directory before continuing.
Next you select the packages to be installed. If this is an upgrade, do not install the “Database Template”. If you also plan to run the Hydro-Met Analysis System, select the bottom two components.

On Windows systems, the install procedure will create shortcuts. Please select the shortcut group, or choose the default, “DcsTool”.

Please make sure you follow the dialogs all the way through to the ‘Done’ button.
2.4 Setting your Runtime Environment

The executable programs in the toolkit are stored in the “bin” subdirectory under the installation. You should add this directory to your system PATH variable.

Previous toolkit releases also advised you to set the $DCSTOOL_HOME variable. This is no longer necessary and you can remove this from your environment.

2.4.1 Setting the PATH Variable on Windows 2000 and XP

To set environment variables on Windows, do the following:

• Click: Start - Control Panels - System
• Click the “Advanced” Tab
• Press Environment Variables
• In the “System Variables” section, modify the Path variable.
• Add “c:dctool\bin;” to the beginning of the path string.

2.4.2 Setting the PATH Variable on Linux and Unix

You will need to modify your PATH variable to include the ‘bin’ directory under the toolkit installation. For example, if you installed the toolkit at ‘/home/lrgs/DECODES’, then add following to your login environment files (assuming you use sh, ksh, or bash).

```
PATH=/home/lrgs/DECODES/bin:$PATH
export PATH
```

For C shell, use the setenv command.
2.5 Activating the Toolkit License

Note: The Toolkit Scheduler will not run without a valid license!

The DCS Internet Toolkit is licensed to a particular machine. To obtain a license you will need the exact host name of the machine where the toolkit will run. You can obtain this by running the command:

```
hostname
```

This command works on both UNIX and Windows (in a Command-Prompt DOS window). Supply this name to Ilex Engineering for a permanent key. Caution! The toolkit will check the keyed name against the value returned by the hostname command. Supply the name and/or domain name exactly as returned by the command.

If you do not have a serial number, you may request an evaluation license that is good for 60 days. You will be provided with a 12-character activation key.

Start the toolkit.

- Windows: Double-click on the dcstool_start.bat file in the toolkit bin directory.
- Unix: Type the command “dcstool_start”.

Click on the bottom button, labeled “DCS Toolkit Setup”. Then click on the “License” tab. In the space provided, type (or better -- copy/paste) your activation key and press the “Activate” button. Be sure to type the key exactly as it is provided to you, including any plus, minus, slash, or equals signs. The system will respond with your valid serial number and the date of the license expiration, if any.

2.6 Set up the Toolkit for a SQL Database

If you plan to use the Tempest Hydro-Met Analysis System: Stop! See the installation in the Hydro-Met Analysis System User Guide. The analysis system installs a database that includes the DECODES and DCS Toolkit Schema. The following section is for installing the DCS Toolkit and DCP Monitor Schema in stand-alone fashion.

2.6.1 Install PostgreSQL

For Red Hat Linux Systems, this is very easy. When you install the OS, just select PostgreSQL Database Server in the package list.

For windows system, point your web-browser to http://www.postgresql.org. Then find the latest download for your operating system. At the time of this writing it is a file called “postgresql-8.3.3-1.zip”. Unzip this file to a temporary directory and run the installer contained therein.

Then select the Language you want to use in the database. The default package selections will suffice.

The Service Configuration screen is then shown. We recommend that you leave Account name as ‘postgres’ and allow the installer to create this account if necessary.
In the next screen (Error! Reference source not found.) you identify a database user that will have administrative privileges. We recommend that you keep the default ‘postgres’ and that you enter a good password.

If this is a stand-alone system where all software components will run on a single machine, then the defaults will suffice. If, however, you will want to connect to this database from other machines, then check the box to ‘Accept connections on all addresses’.

The default procedural languages and ‘contrib-modules’ will suffice.
2.6.2 Configure the PostgreSQL Server

Edit the file ‘postgresql.conf’.

- On Windows this will be in C:\Program Files\PostgreSQL\8.3\data
- On Linux, this is found at /var/lib/pgsql/data.
- On Solaris, this is found at /var/postgres/8.2/data

Change the time-zone to UTC:

    timezone=UTC

Note that this has nothing to do with what time-zones are displayed. But rather it tells the system to store all dates & times in UTC. This is strongly recommended.

If this is a stand-alone system where all components reside in a single box, then the default listening address of ‘localhost’ is fine. On the other hand, if you plan a distributed system where different components run on different computers, then change:

    listen_addresses = *

In most cases, the default listening port of 5432 will be fine.

After changing these items, stop & start the PostgreSQL service, which you can do through the new “pgAdmin III” tool.

2.6.3 Create DECODES Administrative Database User

The next step is to create a PostgreSQL user for DECODES database administration. This account will be used to create the DECODES database, and to create other user accounts. Note that this is different from a Unix (or Windows) user account.

We recommend that you create a user account called “decodes_adm”. Some of the scripts (and this manual) assume that this user has been created and has complete permissions on the DECODES database.

Login as root, and execute the following commands

    su - postgres
    createuser -P decodes_adm

This will ask if you want to allow this user to be able to create databases (answer yes) and to create new users (answer no).

You are also asked for a password. Choose a good password and remember it.
2.6.4 Configure DECODES to use SQL

Start the toolkit and click on the DECODES Properties tab.

- For editable Database set Type to SQL.
- Set Location to an URL as shown. This contains the hostname and the database name, which we call “decodesedit”
- Press the DB Password button and type the name of the user and password that you created in the previous session.

2.6.5 Create the DECODES Database Schema

After you have configured database authentication (as described in section Error! Reference source not found. for PostgreSQL, and run the setDecodesUser account (as described in section Error! Reference source not found.), and set your decodes.properties file (as described in section Error! Reference source not found.); then you are ready to create and initialize the DECODES database.

Edit the file bin/createDecodesDb.sh. Near the beginning of the file you will see where the variable DBNAME is defined. The default name is “decodesedit”. If you want a different name, change it.

Run the following command:

```
createDecodesDb.sh
```

This script will do several things:

- Create the new database with the name you supplied.
• Use the SQL interpreter to define the tables in the database.
• Runs the dbimport Java program to populate the database with enumerations, EU conversions, and data types.
• Initialize several tables from XML files found in the directory “edit-db”.

You will see several messages printed to the screen as the script does its work. If any errors occur, descriptive information will be printed. If errors are encountered during the ‘dbimport’ phase of the script, additional information will be printed in the file “util.log”.

If you are running the script a second time, you may see error messages from the create-database or table-definition phase saying that the database or table already exists. It is safe to ignore these warning messages.

If you really want to delete the database and start with a clean slate, for PostgreSQL, issue the command:

```
dropdb decodesedit
```

CAUTION: This command will delete the database and all data in it!
3 Running the Toolkit Graphical User Interface (GUI)

To start the toolkit GUI on Windows, double-click the file “dcstool_start.bat”, which can be found in the “bin” directory under the installation. You may want to make a short-cut to this on your desktop or start-menu for easy access in the future.

On Linux/Unix, type the command “dcstool_start” at the command prompt.

The top-level “Launcher” for the toolkit is displayed. From here you can select the toolkit components. If you installed the Tempest Hydro-Met Analysis System you will see the full menu as shown below. If not you will only see the “DCS Toolkit Components.”
4  DCS Toolkit Setup

Select “DCS Toolkit Setup” from the Launcher and the following screen is shown. The setup is divided into four different tabs. Each will be explained below.

Please note the setting for ‘Command to launch plain text editor. For windows systems, we recommend the setting shown above, which will display text files inside the Windows ‘notepad’ program.
Configuration

The Configuration tab contains several variable that control toolkit actions. These are:

**DCSTOOL_HOME:** The value of this environment variable is shown for your convenience, but you cannot change it without re-installing the toolkit. *The toolkit scripts have the install location embedded within them. You cannot move the scripts or rename the directory.*

**Directory for Retrieval Process Files:** This is usually the “procs” subdirectory under the `$DCSTOOL_HOME` setting. You may change it to any other complete path. The scheduler will not run retrieval processes that are placed in any other directory than the one shown here.

**Command to Launch Plain Text Editor:** A plain text editor is used to view certain ASCII files (e.g. log files). Type the command here to launch your favorite text editor. Note “notepad” is recommended for Windows machines, “gedit” for Linux.

**Command to Launch Web Browser:** A Web browser is used to view help files. The default is “mozilla”, which can be downloaded for most operating systems from www.mozilla.org.

**DECODES Database Selection:** Most users have a simple DECODES installation with a single “editable” database. If this is the case for you, check the box labeled “Use Editable DECODES Database.

**Directory for LRGS-Style Network Lists:** The toolkit can use either LRGS-style network list files, or Network Lists Records from the DECODES database.

### 4.1 Software Updates

The “SW Updates” tab is shown below. Periodically you may want to visit this screen and press the button labeled “Check for Updates”. The system will check on the Ilex Engineering, Inc. web site for any updates that are available.

If an update is available, the screen will provide instructions on how to proceed.
4.2 License Activation

The License tab is shown below. Enter your activation key exactly as provided and press the Activate button. The results of activation are shown in the window.

You only need to do this after first installing the toolkit, or when replacing a temporary evaluation key with a permanent one.

If you have trouble with activation, make sure you entered the key exactly as provided. Upper and lower-case characters are distinct. Also make sure you provided your exact hostname to ILEX for generating the key (again, upper/lower-case matters). The key is based on the response to the ‘hostname’ command on your system.

●
4.3 **DECODES Properties**

The final tab in the setup dialog is shown below. These settings control how your DECODES database is accessed. See DECODES User Guide for details.
5 Retrieval Processes

A Retrieval Process is the main workhorse of the toolkit. This section explains the theory behind retrieval processes, and how to create, edit, stop, start, and delete them.

5.1 What is a Retrieval Process?

A retrieval process does the following:

- Connects to one or more DCP Data Servers that you specify.
- Retrieves data that you specify by network list, DCP address, time range, etc.
- Optionally decodes the raw data into time-tagged engineering units.
- Formats the data and places it in a file on your local hard disk.
- Optionally executes a command-line that you specify after each file is created.

You can have any number of retrieval processes. As examples, the toolkit comes with three retrieval processes ready-to-run:

- “sample-10-min-chan34” - Runs every 10 minutes and retrieves the last 10 minutes-worth of data from GOES channel 34.
- “sample-interactive” - Runs on command to retrieve the last hour’s worth of data on GOES channel 61
- “samples-realtime-chan65” - Runs continuously to retrieve data from GOES channels 61, 63, 65, 67, and 69.
5.2 The Retrieval Process Scheduler

Retrieval processes are run in the background by the process called the “Scheduler”. When you start and stop your retrieval processes, the GUI screens send messages to the background scheduler. Obviously the scheduler must be running for this to be successful.

You can start the scheduler as a background process on the screen described below in section 5.4. You can also run the scheduler as a daemon (Unix/Linux) or service (Windows). See the accompanying DCS Toolkit Configuration Guide for instructions on setting up the background service.

5.3 The Retrieval Process Monitor and Control Screen

Click the “Retrieval Processes” button on the Launcher and the following screen is displayed:

From this screen you can view, control, modify, and create your retrieval processes. Retrieval processes are run in the background by the Scheduler. First, make sure that the scheduler is running. The upper left of the screen indicates whether or not the scheduler is currently running. If it is “Stopped”, you can start it with the button to the right.

Starting a retrieval process will have no effect if unless the scheduler is running.

The most recent log messages from the background scheduler are shown in the area at the bottom of the screen. More severe messages will be shown in yellow or red.

You can view the complete scheduler log file for the scheduler if you suspect that problems have occurred. Pressing the “View Scheduler Log” button will bring up a text editor with a snapshot of the current log.

If the text editor does not appear, check the command that you entered for the text editor, as described in section 0.
5.4 *Creating or Editing Retrieval Process*

To create a new Retrieval Process, press the “New” button on the bottom right. To edit an existing Retrieval Process, select it from the list and press “Edit. In each case the editor window, as shown below, is displayed.

![DCS Retrieval Process: sample-retrieval-proc.proc](image)

Note the tree tabs along the top:

- **General:** Here is where you define the inputs and outputs, and when the process will be run. This example will be run every half-hour at 5 minutes after the hour.
- **Search Criteria:** Here is where you specify which DCP messages you are interested. You can select by time range, DCP Address, network list, GOES channel, etc.
- **Advanced:** Here is where you control various advanced features.

Make changes to the retrieval process on this screen. To save your changes, select “Save” or “Save As” from the File menu.
Where to Get Data? (General Tab, Upper left of screen)

Select up to 3 data sources (primary and two backups) from the pull-down menus. The toolkit comes with pre-configured data sources for the two public DCP Data Servers at Wallops, VA. These servers are operated by the National Oceanic and Atmospheric Administration (NOAA).

To create additional data servers, see the instructions on creating “data source” records in the DECODES User Guide. To connect to a server, you typically need to provide a valid user name.

Search Criteria are used to specify to the server which data you want to retrieve. Search criteria are explained in detail in section 9 below.

When to Run? (upper right)

When do you want the scheduler to run this retrieval process? There are four options: Interactive, Continuously (real-time), every $N$ hours, or every $N$ minutes. The selection you make may affect the LRGS Since/Until time range that you entered in your search criteria. The following table provides details on each selection:

<table>
<thead>
<tr>
<th>Selection</th>
<th>Description</th>
<th>LRGS Time Range</th>
</tr>
</thead>
<tbody>
<tr>
<td>Continuously (Real-Time)</td>
<td>The process is run immediately after you start it. The search criteria control the time-ranges (if any). If there is no UNTIL time specified in the search criteria, the process will run indefinitely.</td>
<td>No change. Search criteria is exactly as you entered it.</td>
</tr>
<tr>
<td>Interactive</td>
<td>The process is run once immediately after you start it. It stops when the server reports that you have received all the data currently in storage.</td>
<td>Adds: LRGS_UNTIL: now</td>
</tr>
<tr>
<td>Every $N$ Hours</td>
<td>The process will be run at the specified times. For example, to start every hour at 9 minutes after the hour, enter “9” for number of hours, and “00:09:00” for Start time. The process will terminate when the server reports that you have received all the data currently in storage</td>
<td>Adds: LRGS_SINCE: now - $N$ hours 5 minutes LRGS_UNTIL: now</td>
</tr>
<tr>
<td>Every $N$ Minutes</td>
<td>Just like hourly, but with the capability of running more often.</td>
<td>Add: LRGS_SINCE: now - $N$ minutes 15 seconds LRGS_UNTIL: now</td>
</tr>
</tbody>
</table>
How to Format Output? (middle of the screen)

The retrieval process can run DECODES as a sub-process to produced time-tagged engineering units in a variety of formats. To run DECODES, check the radio button and select the desired output format. Also select the time-zone you want for the sample time-tags.

If you simple want raw, unmodified DCP data, check the radio button for Raw.

Whether or not you decode the data, you can enter delimiters that are placed before and after each message in the output stream.

Where to Put Data? (bottom of the screen)

There are five choices: Pipe to Command, DECODES Consumer, and three separate choices for constructing files containing the DCP message data.

If you select “Pipe to Command Listed Below”, then you must also enter a valid command line. The command will be executed by the Retrieval Process and DCP data will be piped to its standard input.

If you select “DECODES Consumer”, then you must select one of the consumer types from the pull-down list, and then supply an ‘argument’ to be used by the consumer. In the example shown, we are sending data to a “Socket Server Consumer” that listens on port 17009.
Specifying a File Name

There are three options for creating files:

- Construct a separate file for each message.
- Construct a separate file every time the process is started.
- Construct a separate file for each specified time period.

The Filename Template controls the name of the file(s) that will be created. It is called a template because the string may contain the following variables:

- `$PROCNAME` - will be replaced with the name of the retrieval process.
- `$SITENAME` - will be replaced by the site name stored in your DECODES database.
- `$TRANSPORTID` - will be replaced by the DCP address
- `$DCP_ADDRESS` - a synonym for `$TRANSPORTID`, will be replaced by the DCP address.
- `$DATE` - will be replaced with the date & time that the file was created. This may be optionally followed by a format specification in parentheses. For example:
  - `$DATE/yyyyyMMdd-HHmmss)
  - `$DATE(yyyyyMMdd-HHmmss)

If you create a separate file for each message, be sure to add either `$SITENAME`, `$TRANSPORTID`, or `$DCP_ADDRESS` to the template. This will avoid the case where multiple messages with the same time-stamp would overwrite each other.

The DATE value will be formatted according to the template you put in parentheses. The possible formats are extensive. They are described in the Java doc page for the class SimpleDateFormat, which can be found on-line at:

http://java.sun.com/j2se/1.4.1/docs/api/java/text/SimpleDateFormat.html

Make sure that you have no embedded spaces in the filename template. This would result in an illegal filename on most operating systems.

A final word of caution: sometimes you need to enclose the above names in curly-brackets. The substitution mechanism will work on names containing underscore and period. Example:

Wrong:  

Wrong:  

Right: 

In the first case, the string it tries to substitute is ‘$TRANSPORTID_’ , which of course won’t work. The same applies if a period follows the name.
5.5 Starting & Stopping Retrieval Processes

To start a process, open the Monitor and Control screen. Select the desired process and press the “Start” button along the bottom. In the screen shown below, we have started the first process.

![Retrieval Process Monitor and Control](image)

Remember: A retrieval process will not actually run (even if Enabled=yes) unless the scheduler is running!

5.6 Monitor a Running Retrieval Process

The columns of the Monitor & Control screen have the following meaning:

- **Enabled?** Either “yes” or “no” depending on whether you have pressed the “Start” button. Some processes (interactive) will complete and then disable themselves. Real-time or periodic processes stay enabled until you disable them via the “Stop” button.
- **Status** shows the current status of the individual process. This may be “Running” if the process is actively running, “Waiting” for periodic processes that are scheduled to run in the future, or possibly some error condition if a process has run into trouble.
- **Last Retrieval** is the time that the process last retrieved data. The time is shown in UTC in the format DDD HH:MM:SS (DDD is day of year).
- **Msgs/Run** is the number of messages retrieved by the process in the current (or last) run. For periodic processes, this is reset to zero every time the process restarts.
- **Msgs/Today** is the total number of messages retrieved in the current day (UTC).
- **Next Run** shows the time that the process is next scheduled to run. This applies only to periodic processes.
- **Last Svr Used** is the name of the server from which data was last (or currently) received.
6 The DECODES Database

The DCS Internet Toolkit is tightly integrated with DECODES. Even if you don’t plan to decode data, the DECODES database is used to store information about your data servers.

You can start the DECODES database editor from the top-level Launcher screen.

For complete information on the editor see the DECODES Users Guide, available as a free PDF download from http://www.ilexeng.com. This section contains information on using the editor to create data source records only.

6.1 Data Server Records

Start the DECODES database editor from the Launcher. Click on the “Sources” tab along the top. The screen shows as follows:

This screen shows two Data Source records. Both are of type “LRGS” which means that this record describes how to pull data from an LRGS (or from DAPS) using DCP Data Service (DDS).

You can create new server records, or open existing ones using the buttons along the bottom. Opening the first record, labeled “cdadata.wcda.noaa.gov”, results in the following screen:
In the open tab, you see the name and type of the data source. Along the right you see a set of properties that control this data source.

Accepted properties in an LRGS Data Source are as follows:

- **host**: The host name or IP Address of the LRGS system to connect to. (Optional, If missing, the name of the data source object is used.)
- **port**: Port number for this LRGS’s server. (Optional, default = 16003)
- **username**: registered user on the LRGS server (required)
- **password**: Some LRGS servers are configured to require passwords. If this is the case, you will need to enter the password here. *Warning! The password will be stored in clear text in the DECODES.*
- **single**: (Default=false) The newer LRGS servers have a new feature whereby many DCP messages can be returned for a single request. By default, DECODES will use this feature if the server supports it. To force the old (single message per request) behavior, add a property “single” with a value of either “on”, “true”, or “yes”.
- **sendnl**: (Default=true) – Old DRS servers do not support network list transfers. Set this to false when connecting to such servers. The data source will then assume that the network lists are already loaded on the DRS. You must then transfer the list using some other mechanism (e.g. FTP) prior to running the routing spec.
- **response.timeout**: (Default=60 seconds) This is the number of seconds to wait for a response from this server.
6.2 Running DECODES Programs from the Command Line

DECODES contains many utilities that can be run from the command line. A list is provided below. See the DECODES User Guide for complete information on each command.

dbedit Start the DECODES Database Editor.

dbexport Export your entire DECODES database to a single XML file (for backup or exchange).

dbimport Import a DECODES XML file into your database.

dbinstall Copy the ‘production’ elements from your editable to your production (installed) database.

emitimport Import SDF File from by EMIT or legacy C-DECODES programs.

lrgsgui Start the LRGS CORBA GUI to view real-time status, do system administration, view messages, etc.

msgaccess Start the LRGS Message Browser

nl2lrgs Converts a network list in your DECODES database into an LRGS-style Network List File.

pxport Export a group of platforms (by name, network list, or all) to an XML file for backup or exchange.

rs -e Start a routing spec (The -e argument means to use the editable database).

setDecodesUser For SQL databases, this needs to be run once prior to accessing the database.
7 Maintaining Network Lists

A “network list” is a named list of data collection platforms. Your retrieval processes tell the servers which platforms it is interested in by sending it a network list.

Network lists are stored in ASCII files in the “netlist” subdirectory under you toolkit installation. By convention, these files will have a “.nl” extension.

Press the “Network List Maintenance” button on the toolkit Launcher screen. The following shows an open network list.

As you can see, for each entry, you enter the NESDIS 8-hex character ID, a mnemonic name, and a description.
# Browsing for DCP Messages

The Message Browser screen is shown below.

In the upper left quadrant you specify connection information:

- **Host Name** is either a fully-qualified domain name, an alias, or an IP address specifying the LRGS host you want to connect to.
- **Port** is a numeric TCP port number. The LRGS DCP data server uses port 16003 by default. Some legacy servers are known to use port 9999.
- **User Name** specifies your ID for connecting to the host. This name must be recognized by the server in order for you to be able to access data.

The middle-left section of the screen is where you specify search criteria. You can specify the name of a search criteria file. This file will be downloaded when you first try to display a message.

The "Select" button brings up a file-selection dialog for you to navigate to, and select a file. Once selected, you can press the "Edit" button to bring up the Search Criteria Editor. See section 9 below for details on search criteria, and the editor.

The check-box labeled "Send Network Lists" allows you to specify how network lists are handled. A search criteria file can specify network lists to be used. These lists might already reside on the LRGS in your user directory, or in one of the LRGS directories. If this is the case, leave this box un-checked.
Conversely, you may be using a new network list that only resides on your client machine. If this is the case, check the box. The network lists will be downloaded to the LRGS before the search criteria file is transferred.

When the server encounters a network list name in a search criteria file, it looks in directories in the following order:

- If a complete path is specified, only that directory is searched.
- The current DCP Data User's directory (This is typically a sub-directory under ~lrgs/users).
- The current user's UNIX home directory, if one exists.
- ~lrgs/netlist
- ~lrgs/netlist/remote
- ~lrgs

The lower-left area controls the Display Format for each DCP message. The 'Prefix' string is printed before the message. The 'Suffix' string is printed after the message. When the 'Wrap Long Lines' check box is checked, the horizontal scroll-bar will disappear. Long lines will be wrapped. When un-checked, lines of data will not be wrapped. Rather, a horizontal scroll bar will appear allowing you to view the entire message.

### 8.1 The Message Output Screen

If you press “Save To File” from the Message Browser screen, The Message Output Screen is displayed, as depicted below.

In this window you specify an output file to receive the data specified by your search criteria. The radio buttons along the left allow you to specify what to do if the file already exists.

Press 'Run' to start saving data to the file. The DCP address, time-stamp, and message count will be displayed in the screen along the right.

You can pause output by pressing the 'Pause' button. Press 'Run' to continue.

If you want the window to automatically close when the specified 'UNTIL' time is reached, check the box labeled 'Close When Done'.

![DCP Message Output Screen](image)
9 Search Criteria

A “Search Criteria” file is where you specify the messages of interest. You can specify by time range, DCP, GOES channel, and a few other parameters.

You will likely create and edit your search criteria using the editor described in section 9.1 below. Once, saved, search criteria files are sent to the server, which then returns the matching DCP messages.

Search criteria are stored in text files, which you can also edit manually if desired. The file format is described in section 9.2 in detail.

9.1 Search Criteria Graphical Editor

The search criteria editor screen is shown in Figure 9-1.

![Search Criteria Editor](image)

Figure 9-1: Search Criteria Editor.
9.1.1 Specifying Time Range

There are 3 ways to specify the beginning of the time-range, as shown in the pull-down list under ‘Since’:

- **Now Minus**: allows you to specify an interval relative to the current time. For example, to get the last hour’s data, Select “Now –”, and then select “1 hour” from the drop-down list.
- **Calendar**: allows you to specify a specific time from a calendar, as shown in Figure 9-2. The widget will save the time in the correct format when saving the file.
- **File Time**: Allows you to specify a file. Either type the name or browse to it. The since time will be set to the last-modify-time of the file.

![Calendar-Widget](image)

**Figure 9-2: Selecting times using the Calendar-Widget**

There are 4 ways to specify time the end of the time-range:

- **Real Time**: means that there is no ‘until’ time. The client will continue retrieving data forever or until some type of connection error occurs. As new data arrives on the server it is transferred to the client.
- **Now**: means that the client should stop when the server informs it that all the data currently in storage has been sent.
- **Now Minus**: Tells the server to stop some interval before ‘Now’. You specify the interval in the drop-down list.
- **Calendar**: Specify a particular date/time to stop retrieval.
9.1.2 Specifying Platform(s) by Name, Address, or List

You can specify the platform(s) of interest in 4 ways:

- **DB Netlist**: This is a network list in your DECODES database. Press the button and select a list from the pull-down.

- **File Netlist**: Some people prefer file-based network lists, particularly if they do not use the DECODES database. This button allows you to specify a filename.

- **DCP Address**: Specify a particular DCP address in the pop-up.

- **DCP Name**: The system will attempt to map the name you type to a DCP address by looking it up in your DECODES database. If this fails, it sends the name to the server to see if it can map it.

9.1.3 Specify GOES Channels, Spacecraft, or Baud Rate

Click the “Channel” radio button and press the “Select” button to bring up the dialog shown in Figure 9-3.

![Figure 9-3: Channel-Select Dialog](image-url)
9.1.4 Qualifiers and Other

“Retransmissions” are messages that have been transmitted over DOMSAT more than once. You can specify 3 choices:

- **Yes**: Include retransmissions along with current messages.
- **No**: Filter out retransmissions.
- **Only**: Include only retransmissions. Filter out current messages.

“DAPS Status Messages” are generated by the ground system. These are sometimes called “Abnormal Response Messages”. They describe things like “outside normal time window”, “received on wrong channel”, and other possibilities. See the LRGS manual for a complete list.

For DAPS Status messages, you have the same three choices: Yes, No, or Only.

The two checkboxes in the lower right are new features of the DDS protocol and may not work on the LRGS server you are connected to. The client software will not send these settings to older-version servers.

- **Ascending Time Only**: This forces the server to sort messages in ascending time-order before returning them to the client. Doing this may slow retrieval down.
- **Real-Time Settling Delay**: Due to the way the LRGS performs merging of several data streams, a client retrieving data in real-time may get duplicate copies of the same message. Checking this box will prevent this duplication. The down-side is that it will add small (several second) delay on real-time messages. That is, your client will not be given messages until several seconds after they are retrieved.
9.2 Search Criteria File Format

A search criteria file is a text file containing a series of keyword-value pairs, one per line. By convention, search criteria files should have a “.sc” extension. Each keyword signifies a particular criterion that DCP messages must pass in order to be returned.

Each line begins with a keyword, followed by a colon, followed by a string value. Here are the available keywords:

**DRS_SINCE**
Only retrieve messages that were received after the specified time. *See allowable time formats below.*

**LRGS_SINCE**
Synonymous to DRS_SINCE.

**DRS_UNTIL**
Only retrieve messages that were received before the specified time. *See allowable time formats below.*

**LRGS_UNTIL**
Synonymous to DRS_UNTIL.

**DAPS_SINCE**
Only retrieve messages with a DAPS time-stamp after the specified time. *See allowable time formats below.*

**DAPS_UNTIL**
Only retrieve messages with a DAPS time-stamp before the specified time. *See allowable time formats below.*

**NETWORK_LIST**
The value following this keyword is a network list file. Only retrieve messages whose DCP address is contained in the list. For multiple lists, put multiple lines in the search criteria file, each beginning with the NETWORK_LIST keyword.

**DCP_ADDRESS**
Only retrieve messages with the specified DCP address. To specify multiple addresses, put multiple lines in the search criteria file, each beginning with the DCP_ADDRESS keyword.

**DCP_NAME**
Only retrieve messages with the specified DCP name. Names are mapped to DCP addresses in network list files. See the section below on Network List Files for details.

**CHANNEL**
Only retrieve messages that were transmitted on the specified GOES channel. The value is a number only. The GOES spacecraft identifier (‘E’ or ‘W’) is not necessary.
9.2.1 DRS verses DAPS Time

The DAPS time is the time the message was received by the DAPS system in Wallops, Virginia. This corresponds to the time the message was sent by the originating DCP.

The DRS (a.k.a. LRGS) time is the time the message arrived at the server you are receiving data from. Servers receive their messages in various ways (DOMSAT, GOES-DRGS, Internet, etc.). Every server time-stamps each message as it arrives.

DRS time searches are very efficient. The server can perform a fast binary-search on messages in storage to quickly find the specified range.

DAPS time searches are less efficient because messages are not in DAPS-time order in storage. The server must perform a more-lengthy linear search.

A good approach is to specify both DAPS and DRS time ranges.
9.2.2 Allowable Time Formats for a Search Criteria File

The SINCE and UNTIL values can take one several time formats.

Relative formats start with the keyword “now” and then add or subtract increments. For example:

- now – 20 minutes
- now – 1 day
- now – 1 week 3 days 20 minutes 10 seconds
- now

You can specify an absolute GMT value in one of the following formats.

- YYYY/DDD HH:MM:SS complete specification
- YYYY/DDD HH:MM seconds assumed to be 00
- DDD HH:MM:SS assume current year
- DDD HH:MM seconds assumed to be 00
- HH:MM:SS assume current day
- HH:MM seconds assumed to be 00

You can specify that output should start with the last message you retrieved from a previous session. This is a special value that can only appear in the LRGS_SINCE field. Simply type the word:

- last

The “last” keyword provides an easy way to connect periodically and processes all messages that have arrived since your last session. Simply connect periodically and use the time range:

- LRGS_SINCE: last
- LRGS_UNTIL: now

The server tracks the last message received by each user. So if you plan to use “last”, make sure that no one else is using your DDS account.
10  Start Toolkit Programs from the Command Line

The toolkit programs are designed to be started and stopped from the integrated graphical user interface. However, there may be special circumstances to launch from the command line.

All of the programs are implemented in 100% Java. The ‘bin’ directory under your toolkit installation contains many scripts for starting the toolkit, DECODES, and LRGS Client programs. In the bin directory, files that end with “.bat” are DOS batch scripts for windows systems. Scripts without an extension are for Unix systems.

<table>
<thead>
<tr>
<th>Command</th>
<th>Application</th>
</tr>
</thead>
<tbody>
<tr>
<td>dbedit</td>
<td>DECODES Database Editor</td>
</tr>
<tr>
<td>dbexport</td>
<td>Export DECODES Database to XML file</td>
</tr>
<tr>
<td>dbimport</td>
<td>Import XML file to DECODES database</td>
</tr>
<tr>
<td>dbinstall</td>
<td>Copy approved elements to your production DECODES database.</td>
</tr>
<tr>
<td>decj</td>
<td>Low-level script to start toolkit and DECODES programs with appropriate Java arguments.</td>
</tr>
<tr>
<td>dcstool_start</td>
<td>Starts the Toolkit Launcher</td>
</tr>
<tr>
<td>emitimport</td>
<td>Import Legacy EMIT-formatted SDF files to DECODES.</td>
</tr>
<tr>
<td>lrgsgui</td>
<td>Starts the LRGS Graphical User Interface</td>
</tr>
<tr>
<td>msgaccess</td>
<td>Starts the DCP Message Browser</td>
</tr>
<tr>
<td>nl2lrgs</td>
<td>Convert DECODES network lists to LRGS-style network list list files.</td>
</tr>
<tr>
<td>pxport</td>
<td>Export DECODES platform records to XML file.</td>
</tr>
<tr>
<td>rpmc</td>
<td>Starts the Retrieval Process Monitor and Control screen directly.</td>
</tr>
<tr>
<td>rs -e</td>
<td>Runs a DECODES Routing spec</td>
</tr>
<tr>
<td>sched_start</td>
<td>Starts the Toolkit Scheduler</td>
</tr>
<tr>
<td>setDecodesUser</td>
<td>Set username for access to SQL DECODES database.</td>
</tr>
</tbody>
</table>

The DEBUG Level Argument:
Almost all of the above commands can take a debug-level argument to control the verbosity of messages sent to the run-time log. As an aid to troubleshooting, you may sometimes want to start programs from the command line with either -d1, -d2, or -d3 (3 is the most verbose).

10.1  The ‘getDcpMessages’ Utility
A command line program called getDcpMessages is included in the toolkit. You can use this program to retrieve DCP messages from any DDS server. We provide a script in the bin directory called ‘getDcpMessages’ (for Windows, ‘getDcpMessages.bat’) to start the program.
Run the program as follows

```
getDcpMessages -u username options...
```

The only required argument is “-u username”. Supply a valid user name on the server you are connecting to.

Options can be any of the following:

- **-p port** Numeric TCP Port. Default is 16003.
- **-h host** Supply a host name or IP address. Default is “localhost”.
- **-u username** The user name that must be known to the DDS server.
- **-P password** An optional password for an authenticated connection to the server.
- **-f searchcrit** The name of the search criteria file to send to the server. If this argument is omitted, no search criteria file will be sent, this causes the server to send all messages currently in storage, which is probably not what you want.
- **-b before** Specifies a string to be output before each message
- **-a after** Specifies a string to be output after each message
- **-n** Causes a newline to be output after each message
- **-v** (Verbose) – Causes various status information to be printed while running.
- **-d level** Sets the debug level: 0 (no debug messages), 1, 2, or 3 (most verbose)
- **-l logfile** Name of log file where debug messages are sent (default=stderr)
- **-t seconds** Timeout value: Number of seconds to wait for a message from the server before exiting.
- **-s** (single) Use this option to force the client to retrieve messages from the server one-at-a-time. By default, if the server is protocol version 4 or higher, the client will attempt to retrieve DCP messages in 80KB blocks. (This is much more efficient.) Hence, only use this argument in trouble-shooting scenarios.

The **before** and **after** strings can contain control and binary characters by using Unix-style escape sequences.

The search criteria file format is described in section 9.2.
11 DCS Toolkit Extensions to DECODES

The toolkit contains several extensions to the standard DECODES Package.

11.1 Output Formatters

We have added several output formatters at the request of our clients. The sections below describe each format including examples of the output, and properties that you can use to control various aspects of formatting.

11.1.1 Output Format: SDML-1 (Site Data Markup Language)

Site Data Markup Language is an XML implementation for exchanging decoded site data. Commercial Packages such as Marta Systems’ Yosemite support the real-time ingest of data that has been formatted according to SDML-1:

```xml
<sitedata>
  <sitename>TYPE=VALUE</sitename>
  <LatLonElev> lat, lon, elev </LatLonElev>
  <description> ... </description>
  ... other site-level parameters
  <sensordata>
    <sensorname>xxx</sensorname>
    <sensornumber>1</sensornumber>
    <datatype>standard=xxx</datatype>
    <units>xxx</units>
    <property>name=value</property>
    ... etc. a sensor may have multiple ‘properties’.
    <time>YYYY MM/DD HH:MM:SS</time>
    <sample>value</sample>
    ... above could be use for individual samples, or...
    <series>
      <interval>#seconds</interval>
      <values> v1, v2, v3, ... </values>
    </series>
  </sensordata>
  ... etc. for each sensor defined for this station.
</sitedata>
```

Notes on SDML-1:

- Only tags and content are used. No attributes.
- The Retrieval Process will output a series of <sitedata> records. Each record contains the data for a single message from a single site.
- The <sitename> record contains the DECODES site name. Every site in DECODES can have multiple names. The ‘preferred’ name will be used. The record will contain the name-type, and equals sign, and the name value. Examples:
  ```xml
  <sitename>NWSHB5=HOMN5</sitename>
  <sitename>ICAO=D1KX</sitename>
  ```
• The <LatLonElev> record contains the latitude, longitude, and (future) elevation values stored in your DECODES site record. If these values are missing, they will be replaced by a single ‘M’. Lat/Lon values should be represented as a floating-point decimal number of degrees. Negative numbers are used for west longitude. Elevation should be feet above mean sea level. Examples:
  <LatLonElev>34.401,-119.18,M</LatLonElev>

• <datatype> records contain STANDARD=DATATYPE pairs. Examples:
  <datatype>SHEF-PE=HG</datatype>
  <datatype>EPA=00065</datatype>

• Properties are used for a variety of purposes within DECODES. All of the properties defined for each sensor will be listed.

• Time values will be in the format shown and must be UTC. The time is the time-stamp for any following <sample> records; or it is the start-time for any following <series> records.

• If only a single data value is available, a <sample> record will contain the value as a floating-point decimal number. Example:
  <time>2003 08/25 12:00:06</time>
  <sample>13.7</sample>

• If multiple data values are present but they are not a regular time series. Multiple pairs of <time> <sample> records may be given. Example:
  <time>2003 08/25 12:00:06</time>
  <sample>13.7</sample>
  <time>2003 08/25 12:11:06</time>
  <sample>13.8</sample>
  <time>2003 08/25 12:32:14</time>
  <sample>13.3</sample>

• If multiple values from a regular time series are available, the <time> record represents the time-stamp of the first value in the series. An <interval> record will indicate the number of seconds between values (which may be negative). A single <series> record will contain comma-separated values.
11.1.2 **Campbell Scientific Comma-Separated-Value Format**

A special formatter has been prepared that uses the Campbell Scientific CSV (Comma-Separated-Value) format compatible with their RTDM analysis product.

This format is as follows:

\[ ID, YYYY, DDD, HHMM, Value1, Value2, \ldots \]

where:

- ID is an identifier field (see below)
- YYYY is the year
- DDD is the Julian day-of-year.
- HHMM is the time of day (using the time-zone you specified)
- Value1, Value2\ldots, are the samples for each sensor.

The ID field is normally the site name. You can override this to be anything you want by doing the following:

- Start the DECODES Database Editor
- Go to the Platforms tab and open the desired platform record.
- In the Platform Sensor area, select sensor number 1 and press the Sensor Properties button.
- Add a property with the name **CSVID** and the desired value.
- Save this record. When you run your retrieval process, the CSVID value you specified will appear in the left-most column of the output for this platform.

This formatter will accept the following properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>YearFormat</td>
<td>String</td>
<td>yyyy</td>
<td>Format for the year column (Java SimpleDateFormat)</td>
</tr>
<tr>
<td>DayFormat</td>
<td>String</td>
<td>DDD</td>
<td>Format for the day-of-year column (Java SimpleDateFormat)</td>
</tr>
<tr>
<td>TimeFormat</td>
<td>String</td>
<td>HHmm</td>
<td>Format for the time column (Java SimpleDateFormat)</td>
</tr>
<tr>
<td>IncludeSiteName</td>
<td>Boolean</td>
<td>True</td>
<td>Set to false to have the site name (column 1) excluded.</td>
</tr>
</tbody>
</table>
11.1.3 Alberta Environment “Alta-Loader” File Format

A special formatter has been prepared for Alberta Environment. This format is called “alta-loader” in the pull-down list.

In this format, each sample is specified on a separate line, with the following format:

```
NNNNNNNN YYYYMMDD HHMMVVVVVVVVSSSSR
```

… where

- ‘NNNNNNNN’ is the station name, left justified and truncated if necessary to 8 characters.
- ‘YYYYMMDD HHMM’ is the date/time stamp of the sample, truncated to 1 minute resolution.
- ‘VVVVVVVV’ is the sample value.
- ‘SSSS’ is the sensor name, left justified and truncated if necessary to 4 characters.
- ‘R’ is the literal character ‘R’, and is used as a record terminator.

At the end of the file, the formatter will place the following literal trailer:

```
END      12340101 000012345678XXC
END      123400000000123456788SSSSC
```

11.1.4 Kisters Wiski ZRXP Format

This formatter creates an output in the Kisters ZRXP format, suitable for importing data into their WISKI product. The following is an example of the output:

```
#REXCHANGE|TRAM5|HG|*|CNAMESTAGE|*|CUNITFEET|*
20050523133000 12.05
20050523134500 12.06
20050523140000 12.06
20050523141500 12.07

#REXCHANGE|TRAM5|PC|*|CNAMEPRECIP|*|CUNITINCHES|*
20050523133000 7.70
20050523134500 7.70
20050523140000 7.70
20050523141500 7.70
```

Each parameter on each platform is given its own header line, followed by a separate line for each sample, with a complete time-tag and value.

The header line has some constant fields and some data filled in from the platform data:

```
#REXCHANGE \text{sitename} \text{paramcode} \text{|*|CNAME} \text{sensorname} \text{|*|CUNIT} \text{units} \text{|*|}
```

By default, the “sitename” will be the NWSHB5 (National Weather Service Handbook 5) name for the site. If you want to use a different name, add a property called “siteNameType” and set its value to the name type you want to use.

By default, “paramcode” will be the SHEF code assigned to this sensor. If you want to use a different code, add a property called “dataTypeStd” and set its value to the code type you want to use.
11.1.5 Raw Formatter

This formatter simply outputs a raw message (that is un-decoded).

It supports the following Boolean property:

- noStatusMessages – value: ‘true’ or ‘false’. If true, then the formatter will skip DAPS status messages.

Use the consumer’s properties to place delimiters before or after each message. For example, if you’re using the ‘pipe’ consumer, the properties ‘consumerBefore’ and ‘consumerAfter’ will place special strings before and after each message in the output.

11.1.6 Pi Historian Formatter

This format allows you to ingest data into the Historian product. The format for a station called “montg” with two sensors called “reslev” and “stage” would look as follows:

```
[Data],
Tagname,TimeStamp,Value
montg_reslev.F_CV,06/16/2006 09:45,75.62
montg_reslev.F_CV,06/16/2006 09:30,75.62
montg_reslev.F_CV,06/16/2006 09:15,75.61
montg_reslev.F_CV,06/16/2006 09:00,75.6
montg_stage.F_CV,06/16/2006 09:45,75.62
montg_stage.F_CV,06/16/2006 09:30,75.62
montg_stage.F_CV,06/16/2006 09:15,75.61
montg_stage.F_CV,06/16/2006 09:00,75.6
```

The first two header lines always appear as shown above. The data lines have three comma-delimited fields:

- A data ID field constructed with site-name – underscore – sensor name, followed by the constant string “.F.CV”.
- The date and time in the format: “MM/DD/YYYY:HH:MM”.
- The data value.
11.1.7 Vaisala "METMAN" Formatter

Add the following entry to the enumeration for Output Formatters (using the ‘redit’ program):

- Mnemonic: metman
- Full Name: Vaisala METMAN Formatter
- Java Class Name: com.ilexeng.decodes.consumer.MetmanFormatter

Data in METMAN format appears as follows:

```
(S:RONM8;D:070131;T:150000;US:1.00000 ;UD:230.000 ;TA:20.0000 ;TX:20.0000
 ;XR:100.000 ;VB:12.7000 ;ZM:26.5000 ;UR:319.000 ;UP:4.00000 ;RN:2.00000 )
(S:RONM8;D:070131;T:152000;PC:6.58000 )
(S:RHFA1;D:070131;T:140000;PC:44.9700 ;US:3.00000 ;UD:87.0000 ;TA:20.0000
 ;TX:22.0000 ;XR:62.0000 ;VB:14.5000 ;ZM:12.4000 ;UR:93.0000 ;UP:12.0000 ;RN:215.000
 )
```

Each line contains a single time-slice of data for a single platform. In this example, “RONM8” and “BHFA1” are platform names. The Date format is YYMMDD. The time format is HHMMSS. Following the date & time is a data-type code and value for each sensor.
11.1.8 SFWMD XML Formatter

An XML format was prepared for SFWMD (South Florida Water Management District) with a slightly different schema than the one described above in section 11.1.1. Each message in this format appears as follows:

```
<DcpData Dcpaddr="DCP-Address">
  <TimeZone>UTC</TimeZone>
  <SystemTime>2007/10/01-12:19:06</SystemTime>
  <FailureCode>G</FailureCode>
  <SignalStrength>38</SignalStrength>
  <FrequencyOffset>+1</FrequencyOffset>
  <ModulationIndex>N</ModulationIndex>
  <Site usgs="USGS Site Number" nwshb5="Weather Svc Name">
    <Sensor shefcode="HG" usgscode="00065" name="Stage" units="ft">
      <Value time="2007/10/01-11:00:00">13.22</Value>
      <Value time="2007/10/01-10:00:00">13.25</Value>
      <Value time="2007/10/01-09:00:00">13.29</Value>
    </Sensor>
    <Sensor shefcode="PC" usgscode="00010" name="Precip" units="in">
      <Value time="2007/10/01-11:00:00">30.44</Value>
      <Value time="2007/10/01-10:00:00">30.44</Value>
      <Value time="2007/10/01-09:00:00">30.44</Value>
    </Sensor>
    <Sensor shefcode="VB" usgscode="70969" name="Battery" units="v">
      <Value time="2007/10/01-12:00:00">12.6</Value>
    </Sensor>
  </Site>
  <Site usgs="2nd USGS Site Number" nwshb5="Weather Svc Name">
    . . . etc. Data for the next site here.
    (in rare cases it is possible for a single message to have data from multiple sites)
  </Site>
</DcpData>
```

This formatter will accept the following properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>includeQuality</td>
<td>Boolean</td>
<td>False</td>
<td>Include the message quality indicators &lt;FailureCode&gt;, &lt;SignalStrength&gt;,</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>&lt;FrequencyOffset&gt;, and &lt;ModulationIndex&gt;</td>
</tr>
</tbody>
</table>

If this formatter is not a choice in your pull-down list, add it with the reference list editor. Start the ‘rledit’ program, In the ‘Enumerations’ tab select ‘Output Formatters’ and add a new entry as follows:

- Mnemonic: sfwmd
- Full Name: South Florida Water Management Dist XML Format
- Java Class Name: com.ilexeng.decodes.consumer.SfwmdFormatter
Notes on the SFWMD Format:

1. Each message is packed into an enclosing <DcpData> element. The DCP address is provided as an attribute.
2. The Time Zone is provided for interpreting all time stamps.
3. The <SystemTime> element contains the time on the system when this message record was generated. Also each data value is time-stamped.
4. <FailureCode> applies to GOES messages only, and is only included in the output if the “includeQuality” property is set to true. It will be ‘G’ for a good quality message, or ‘?’ for a message that contains parity errors.
5. <SignalStrength> applies to GOES messages only, and is only included in the output if the “includeQuality” property is set to true. Signal Strength will be two ASCII digits and will be in the range of 32 to 57. Signal strength is the implied EIRP, assuming the pilot is a +47 dBm reference.
6. <FrequencyOffset> applies to GOES messages only, and is only included in the output if the “includeQuality” property is set to true. Frequency Offset will be two ASCII characters. The first will be a plus or minus sign. The second will be an ASCII digit 0 through 9, or the capital letter 'A'. The sign indicates that the DCP is transmitting above or below (plus or minus, respectively) the channel center frequency. The digit indicates the amount of the offset in increments of 50 Hz. The character 'A' represents 500 Hz, which is the worst case frequency error that GOES Ground Systems can acquire.
7. <ModulationIndex> applies to GOES messages only, and is only included in the output if the “includeQuality” property is set to true. Possible values are: N=Normal (60° ± 5°), L=Low (≤ 50°), H=High (≥ 70°).
8. Most DCPs have all their sensors at a single site. So normally there will only be one <Site> block within a <DcpData> block. Occasionally however, a DCP may report data from multiple sites. In this case, multiple <Site> blocks will be found.
9. Each <Site> block will have attributes representing the names assigned to the site. The format will be: name-type="name-value". The example shows a USGS site number (name type = “usgs”) and a National Weather Service Handbook 5 (NWSHB5) name.
10. Within each <Site> block you have all the sensors at that site.
11. Each <Sensor> record has attributes representing the data-type, sensor-name, and engineering units. Data types are of the form: standard="code". The example shows a sensor named “Stage”. The SHEF code for stage is “HG”, the USGS Parameter Code is “00065”, and the values are in units of “ft” (feet).
12. Within each <Sensor> block you see time-tagged sensor values.
13. Each <Value> element will have an attribute containing a unique time-stamp. The content of the element will be the parameter value at that time.
11.1.9  Generic Comma-Separated-Value Format

To use the Generic CSV Format, open a terminal window and run the ‘rledit’ program, which you will find in the ‘bin’ directory under the toolkit installation. Then click the ‘Enumerations’ tab and select ‘Output Format’ from the Enumeration pull-down list.

Press the ‘Add’ button and fill in the form as shown in Figure 11-1. Be careful to type the Java class name exactly as shown.

![Figure 11-1: Enumeration for CSV Formatter.](image)

Then select File – Save to DB, and exit the RL Edit Program. Then restart the toolkit.

The Generic CSV (Comma-Separated-Value) format looks as follows

```
ID,DateTime,Value1,Value2,...
```

where:

- ID is the optional station-identifier field (see below)
- DateTime contains the date/time stamp. The default format is: `MM/dd/yyyy,HH:mm:ss`
- Value1,Value2…, are the samples for each sensor.

The ID is optional. To have it omitted, set the ‘IncludeSiteName’ property to false. Note that if you do this you should output one file for each message and then encode the site name in the file-name. For example, to have each filename be the site name plus the time-stamp, set ‘File Name Template’ to:

```
$SITENAME-$DATE(yyMMdd-HHmmss)
```

The ID field is normally the site name. You can override this to be anything you want by doing the following:

- Start the DECODES Database Editor
- Go to the Platforms tab and open the desired platform record.
- In the Platform Sensor area, select sensor number 1 and press the Sensor Properties button.
- Add a property with the name **CSVID** and the desired value.
• Save this record. When you run your retrieval process, the CSVID value you specified will appear in the left-most column of the output for this platform.

Normally, the values will contain every sensor. You can control the contents and order by adding a property called ‘columnDataTypes’. Set this property to a string containing the data-types you want in each column, separated by commas. For example, to have 3 columns containing water level (HG), precipitation (PC) and battery voltage (VB), then set:

\[
\text{columnDataTypes} = \text{HG,PC,VB}
\]

When you set ‘columnDataTypes’, then only the specified data-types will be included in each message, and always in the order specified. If a message does not have a value for one of the specified types, the field is empty. For example, the following message has precip only on the half-hour:

\[
\begin{align*}
&01/22/2008,12:15:00,15.32,,12.8 \\
&01/22/2008,12:30:00,15.32,10.11,12.8 \\
&01/22/2008,12:45:00,15.32,,12.8 \\
&01/22/2008,13:00:00,15.32,10.11,12.8 \\
\end{align*}
\]

This formatter will accept the following properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>TimeFormat</td>
<td>String</td>
<td>MM/dd/yyyy,HH:mm:ss</td>
<td>Format for the time column (Java SimpleDateFormat)</td>
</tr>
<tr>
<td>IncludeSiteName</td>
<td>Boolean</td>
<td>True</td>
<td>Set to false to have the site name (column 1) excluded.</td>
</tr>
<tr>
<td>Missing</td>
<td>String</td>
<td>Empty String</td>
<td>The string you would like used to indicate missing or error-flagged data values. The default is an empty string.</td>
</tr>
</tbody>
</table>
11.2  **Data Consumers**

Recall that a ‘consumer’ *consumes* the data coming out of DECODES. That is, it puts it somewhere (like a file or data stream).

11.2.1  **Socket Server Consumer**

Socket Server Consumer allows you to have a Retrieval Process that sends data over the network to remote clients.

- In the “Where to Put Data” section of the editor, select “DECODES Consumer”.
- In the pull-down list of consumer-types, select “socketserver”.
- In the type-in field to the right, type the port number that you want the server to listen on. The default port number is 1779 if you leave the field blank.

When you start the retrieval process, it will “listen” for client connections on the specified port. When a client connects to that port, data will be streamed to the client in real-time.

Since only real-time data is sent to the client, this feature is only appropriate for continuously-running retrieval processes.

A retrieval process can support up to 2 simultaneous client connections.
11.2.2 Soap Client Consumer

The SOAP Client Consumer will send each DCP Message to a SOAP (Simple Object Access Protocol) Server.

To use this consumer in a Retrieval Process:

- In the “Where to Put Data” section of the editor, select “DECODES Consumer”.
- In the pull-down list of consumer-types, select “SOAP-Client”. If you do not see this option, perform the operations below under “Enabling the SOAP Client”.
- In the type-in field to the right, type the URL of the SOAP server.
- In the “How to Format Output” area, specify DECODES and choose the output format “sfwmd-xml” If you do not see this option, perform the operations below under “Enabling the SOAP Client”.
- Click the Advanced Tab.
- Click ‘Add’ to the right of the Properties panel
- Add a property named ‘soapAction’ containing the SOAP action. This must be an action that the server understands.

11.2.2.1 Enabling the Soap Client Consumer

In order to use the consumer and data formatter for the SOAP Client, you must first add two entries to your reference lists.

Open a DOS window (or terminal on Unix), and CD to the Toolkit directory. Then type:

```
bin\rledit    \n```

or

```
bin/rledit    \n```

- Click the ‘Enumerations’ tab at the top.
- Select the ‘Data Consumer’ enumeration.
- Click ‘Add’ and enter the values shown in Figure 11-2.

![Enum Item Dialog](image)

Figure 11-2: Add SOAP-Client Consumer Reference List Entry.
Then …

- Select the OutputFormat enumeration
- Click ‘Add’ and enter the values shown in Figure 11-3

![Enum Item Dialog](image)

Figure 11-3: Add SFWMD XML Output Formatter Reference List entry.

Be careful entering the values for “Executable Java Class”. They must be entered exactly as shown. Upper/Lower case distinctions matter.

Finally, select File – Save to DB.
11.3 **DECODES Format Language Extensions**

The DCS Toolkit supports the following extensions to the DECODES Format Language.

11.3.1 **OTT Pseudo Binary Bulk-Encoded Messages**

The German manufacturer, Ott, provides DCP transmitters that use pseudo binary for bulk-encoding of the DCP messages.

For other manufacturers, a pseudo-binary sample is typically an 18-bit integer encoded as 3 pseudo-binary bytes. So the series \@@A@@B@@C, would decode to the numbers 1, 2, and 3.

Ott takes a different approach. They build a self-describing binary message using 16-bit integers, and then bulk-encode the entire message in pseudo-binary. The result is that the sensor-values do not line up on byte-boundaries. A single pseudo-binary byte might have 2 bits for one sample, and 4 bits for the next.

The DCS Toolkit extends the DECODES format language to handle this. Your script should contain a single format statement as follows:

\[
\text{ott(firstSensorNum, secondSensorNum, ..., NthSensorNum)}
\]

The OTT message is somewhat self-describing. It contains information about intervals, number of sensors, etc. The arguments tell the software what DECODES Sensor is first in the message, what DECODES sensor is 2\textsuperscript{nd}, etc. For example:

\[
\text{ott(1, 3, x, 2)}
\]

This means:

- 1\textsuperscript{st} message sensor goes to DECODES sensor 1.
- 2\textsuperscript{nd} message sensor goes to DECODES sensor 3.
- Skip 3\textsuperscript{rd} message sensor.
- 4\textsuperscript{th} message sensor goes to DECODES sensor 2.
- Skip any subsequent sensor data in the message.

Also in the script you should assign engineering unit names and conversion coefficients, like you normally do.
11.4  **URL Data Sources**

In DECODES a “Data Source” is used to retrieve raw environmental data for decoding. This chapter explains the enhanced Data Sources provided with the DCS Toolkit. For information on reading from local files, directories, or from an LRGS server, see the DECODES User Guide.

A URL Data Source reads data files over a web connection. The connection is specified by an URL (Uniform Resource Locator). The URL may be specified completely or it may be a template containing parameters such as $DATE or $MEDIUMID.

We have implemented three different types of URL Data Sources. The particulars are described in the following sub-sections.
11.4.1 One-Time Setup for URL Data Sources

Before you can use URL Data Source, run the ‘rledit’ program to create some needed entries. Do this with the commands:

```
    cd $DCSTOOL_HOME
    bin\rledit
```

On the Enumerations tab, select “Data Source Type” from the Enumeration combo box. Then press ‘Add’ and fill out the form with the following values (The java class names must be exactly as shown):

- Mnemonic Value: URL
- Complete Description: Read Data from URL
- Executable Java Class: com.ilexeng.decodes.datasource.UrlDataSource

Now add the entry for URL Template Data Source:

- Mnemonic Value: URLTemplate
- Complete Description: Read Data from URL
- Executable Java Class: com.ilexeng.decodes.datasource.UrlTemplateDataSource

Now add the entry for the Canadian Weather Office:

- Mnemonic Value: CWO
- Complete Description: Canadian Weather Office
- Executable Java Class: com.ilexeng.decodes.datasource.UrlDownloader

![Enumeration Table](image)
Then select the Enumeration ‘Transport Medium Type’ and add:

- Mnemonic Value: other
- Complete Description: Other File Download

And:

- Mnemonic Value: mbfire
- Complete Description: Manitoba Conservation Fire Program File

And:

- Mnemonic Value: tcid
- Complete Description: Transport Canada ID

**Important**: Before leaving the reference list editor, Press “File” – “Save to DB”.
### 11.4.2 Simple URL Data Source

A Simple URL Data Source is similar to a File Data Source. The difference is that the file is specified in an URL rather than a local file name.

**Properties for Simple URL Data Source:**

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>url</td>
<td>String</td>
<td>None</td>
<td>Required URL.</td>
</tr>
<tr>
<td>header</td>
<td>String</td>
<td>None</td>
<td>The type of header contained in the file. Should be one of GOES, NOAAPORT, EDL, NOHEADER, or MBFIRE. This will indirectly determine the transport-medium type.</td>
</tr>
<tr>
<td>mediumId</td>
<td>String</td>
<td>None</td>
<td>Specify if the file contains data from only a single platform. If the file contains multiple messages from potentially different platforms, you must provide a header parser (see below) to parse the file.</td>
</tr>
<tr>
<td>before</td>
<td>String</td>
<td>None</td>
<td>If the file contains multiple messages, this is a delimiter that is to appear at the start of each message.</td>
</tr>
<tr>
<td>After</td>
<td>String</td>
<td>None</td>
<td>If the file contains multiple messages, this is a delimiter that is to appear at the end of each message.</td>
</tr>
<tr>
<td>oneMessageFile</td>
<td>Boolean</td>
<td>False</td>
<td>If true, it indicates that the file contains a single message and that the delimiters are moot.</td>
</tr>
</tbody>
</table>
Simple Example:
The national weather service maintains a site of current observations at:

http://www.nws.noaa.gov/data/current_obs/SITENAME.xml

So for the site KMIB, you would download from the URL:

http://www.nws.noaa.gov/data/current_obs/KMIB.xml

This returns an XML file for the site KMIB with the current observations in it.

You could build a dedicated DECODES Data Source to download this one file like this:

See the property settings. We supply the complete URL and we tell it that there is a single
message-per-file, thus delimiters are not necessary.

We used the special noheader. Parsing a header is not necessary because we are supplying the
medium ID directly. This is associated with transport medium type ‘other’.

Now you will need to have a platform with a corresponding transport-medium record with type
‘other’ and ID ‘KMIB’. Then point it to the decoding script in your configuration.
Example with Multiple Messages Per File:
The Manitoba Conservation Fire Program provides observations for many sites on their web site. We will process the file downloaded from:

http://www.gov.mb.ca/conservation/fire/Wx-Hour/NWhour.txt

This file contains row/column data for many sites. The start of each line contains the station name and time stamp. Subsequent columns contain sensor readings.

Therefore we must treat each line as a separate ‘message’. We will set a start delimiter of \n\040 (linefeed followed by a space) and an end-delimiter of \r (carriage return).

We created a header-parser specifically for this file-type. So set ‘header’ to mbfire’:

Now you need transport medium records in your platform with type “mbfire” and the value containing the station name, as it appears in these files.
11.4.3 Template URL Data Source

The Template URL Data Source allows you to iterate over the stations in a network list. Recall the first example above for National Weather Service XML files. By using the template we will not need a separate data source for each file:

![Diagram of template URL data source](image)

We specified the ‘header’ and ‘onemessagefile’ parameters as before. Instead of URL, we provide ‘URLTemplate’. The template contains the variable `${MEDIUMID}`, which will be replaced by the identifiers in the network list.
11.4.4 Canadian Weather Service URL Downloader

The “CWO” data source was implemented for the Manitoba Hydro Organization. They retrieve data from the Meteorological Service of Canada over the web. The MSC provides a web-service with periodic bulletins containing environmental data. These bulletins are under the base URL:

http://dd.weatheroffice.gc.ca/bulletins/

Under this URL we find a directory for each day, named with the format YYYYMMDD. Under each day-directory, we find sub-directories that are started at the beginning of every hour. For example:

http://dd.weatheroffice.gc.ca/bulletins/20080121/SA/CWAO/20/

Under each hourly sub-directory we find the data files containing the DCP messages. For example:

<table>
<thead>
<tr>
<th>File Name</th>
<th>Date</th>
<th>Time</th>
<th>Size</th>
</tr>
</thead>
<tbody>
<tr>
<td>SACN95_CWAO_212000__WZW_85826</td>
<td>21-Jan-2008</td>
<td>20:05</td>
<td>74</td>
</tr>
<tr>
<td>SACN95_CWAO_212012__WXC_52764</td>
<td>21-Jan-2008</td>
<td>20:16</td>
<td>74</td>
</tr>
<tr>
<td>SACN96_CWAO_212000__WXN_51343</td>
<td>21-Jan-2008</td>
<td>20:02</td>
<td>69</td>
</tr>
<tr>
<td>SACN96_CWAO_212000__WXP_85430</td>
<td>21-Jan-2008</td>
<td>20:02</td>
<td>75</td>
</tr>
<tr>
<td>SACN96_CWAO_212000__WXS_51909</td>
<td>21-Jan-2008</td>
<td>20:06</td>
<td>95</td>
</tr>
<tr>
<td>SACN96_CWAO_212000__WXZ_85881</td>
<td>21-Jan-2008</td>
<td>20:06</td>
<td>70</td>
</tr>
</tbody>
</table>

Viewing a particular file at:

http://dd.weatheroffice.gc.ca/bulletins/20080117/SA/CWAO/20/SACN96_CWAO_212000__WXN_51343

Will return the following bulletin:

SACN96  CWAO  212000
WXN  SA  2000  AUTO8  M  M  M/-06/-17/2707/M/ M 09MM=

This CWO Data Source was tested against the following directories:


Restrictions:

1. Currently the CWO Data Source cannot run in Real Time. Thus, if no “Until Time” is specified it will default to “Now”, meaning current time. We recommend you run a retrieval process periodically under the toolkit.

2. Only one ‘base URL’ can be specified for each retrieval process. Thus run multiple processes for multiple URL paths.
CWO Data Source Properties:

<table>
<thead>
<tr>
<th>Name</th>
<th>Type</th>
<th>Default</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>urltemplate</td>
<td>String</td>
<td>None</td>
<td>Required base-URL. See examples below.</td>
</tr>
<tr>
<td>timepos</td>
<td>Number</td>
<td>3</td>
<td>Word-position in the file name where the file timestamp is found. See examples below.</td>
</tr>
</tbody>
</table>

The “urltemplate” property may contain $\text{DATE}($format$)$ and $\text{MEDIUMID}$ strings. The following templates have one field for the date, and another for the hour:

- \url{http://dd.weatheroffice.gc.ca/bulletins/$\text{DATE(yyyymMMdd)/SA/CWAO/$\text{DATE(HH)/}}$
- \url{http://dd.weatheroffice.gc.ca/bulletins/$\text{DATE(yyyymMMdd)/SM/CWAO/$\text{DATE(HH)/}}$

For details on the format-string inside the $\text{DATE()}$ call, see the Java manual page for the ‘SimpleDateFormat’ class. When run, the retrieval process will traverse each hourly directory within the specified time-range.

The following template also uses $\text{MEDIUMID}$:

\url{http://dd.weatheroffice.gc.ca/bulletins/$\text{DATE(yyyymMMdd)/CS/${MEDIUMID}/$\text{DATE(HH)/}}$

The Medium ID uniquely identifies a platform in your DECODES database. Notice how the medium identifier is part of the directory in the URL. When this happens, the retrieval process will fill in the value from the network list you provide. For each platform, it will construct the complete URL and list the contents.

The ‘timepos’ property is the word-number where the timestamp DDHHMM is found in the file name. Example: for file name: SACN98_CWAO_181100__XVN_92634, the timepos is 3 (181100 - is the timestamp ddHHmm). By default timepos is 3. The delimiter for words is assumed to be a single underscore.

Finally create a data source record in DECODES. Open the database editor. Click the “Sources” tab. Give your data source a name. Select “cwo” for type.

11.4.4.1 DECODES Platform, Site, & ConfigRecords
You must now create or modify your DECODES platform, site, and configuration records.

In the configuration record, add a script that can be used to decode the file contents. Give the script a name like “MSC”

In the platform record, add a new transport medium, with the “tcid” type. Make sure you associate the new script-name with the TCID record. The Medium ID is the unique name that appears as part of the file name in the URL. In the example described above, the medium ID is “XVN”:

SACN98_CWAO_181100__XVN_92634

If this is a site that you already have in your database, use the existing site record. If not, create a new site record and associate it with your platform.

Finally, create a network list of your new medium identifiers. Make sure you select “tcid” in the Transport Medium Type combo box.
11.4.4.2 Create Retrieval Process

Create a new retrieval process. On the ‘General’ tab:

- Select the URL Data Source that you created name as “Primary”.
- Under, ‘When to Run?’ select one of the ‘Every’ buttons and specify the desired period.
- Fill out your desired values for output format and where to put the output data.

On the “Advanced” tab, to the right of “Other Properties”, press “Add”. Add a new property with name “urltemplate”. Give it the appropriate value as described above. Here are the three directories tested:

http://dd.weatheroffice.gc.ca/bulletins/$DATE(yyyyMMdd)/SA/CWAO/$DATE(HH)/
http://dd.weatheroffice.gc.ca/bulletins/$DATE(yyyyMMdd)/SM/CWAO/$DATE(HH)/
http://dd.weatheroffice.gc.ca/bulletins/$DATE(yyyyMMdd)/CS/$MEDIUMID/$DATE(HH)/

Then press “Add” again to create a new property with name “timepos”. Type 3 for value.

On the “Search Criteria” tab it is not important to specify a time range. The retrieval process will fill this in automatically from the run-schedule. You do need to type the name of the network list containing your new transport medium IDs.

For troubleshooting, watch the log messages in the retrieval process log:

$DCSTOOL_HOME/RetProcScheduler.log