Documentation Library

Cascade DataHub™ for Linux and QNX

Version 7.0

Cogent Real-Time Systems, Inc.

January 13, 2011

Cascade DataHub™ for Linux and QNX: Version 7.0

A memory resident real-time database that acts as a hub, providing fast and efficient concentration and distribution of data for QNX and Linux applications.

Published January 13, 2011
Cogent Real-Time Systems, Inc.
162 Guelph Street, Suite 253
Georgetown, Ontario
Canada, L7G 5X7

Toll Free: 1 (888) 628-2028
Tel: 1 (905) 702-7851
Fax: 1 (905) 702-7850

Information Email: info@cogent.ca
Tech Support Email: support@cogent.ca
Web Site: www.cogent.ca

Revision History

Revision 7.0-1 September 2007
Updated TCP connectivity and other functionality to maintain compatibility with Windows DataHubs.

Revision 6.2-1 February 2005
Removed synchronous TCP functionality.

Revision 5.0-1 August 2004
Compatible with Cascade DataHub and Cascade Connect for Windows.

Revision 4.0-1 September 2001
Source code compatible across QNX 4, QNX 6, and Linux.

Revision 3.0-1 September 2000
Moved API section to Cogent C API manual.

Revision 2.2 May 2000
Added Using the Cascade DataHub Viewer, Point Locking and Security.

Revision 2.1 April 1999
Converted from Word97 to DocBook SGML.

Revision 2.0 April 1999
Combined User's Manual with API document.

Revision 1.3 March 1999

Revision 1.2 October 1998

Copyright, trademark, and software license information.

Copyright Notice

Trademark Notice

END-USER LICENSE AGREEMENT FOR COGENT SOFTWARE

IMPORTANT - READ CAREFULLY: This End-User License Agreement ("EULA") is a legal agreement between you (either an individual or a single entity) and Cogent Real-Time Systems Inc. ("Cogent") of 162 Guelph Street, Suite 253, Georgetown, Ontario, L7G 5X7, Canada (Tel: 905-702-7851, Fax: 905-702-7850), from whom you acquired the Cogent software product(s) ("SOFTWARE PRODUCT" or "SOFTWARE"), either directly from Cogent or through one of Cogent's authorized resellers.
The SOFTWARE PRODUCT includes computer software, any associated media, any printed materials, and any "online" or electronic documentation. By installing, copying or otherwise using the SOFTWARE PRODUCT, you agree to be bound by the terms of this EULA. If you do not agree with the terms of this EULA, Cogent is unwilling to license the SOFTWARE PRODUCT to you. In such event, you may not use or copy the SOFTWARE PRODUCT, and you should promptly contact Cogent for instructions on return of the unused product(s) for a refund.

SOFTWARE PRODUCT LICENSE

The SOFTWARE PRODUCT is protected by copyright laws and copyright treaties, as well as other intellectual property laws and treaties. The SOFTWARE PRODUCT is licensed, not sold.

1. **EVALUATION USE:** This software is distributed as "Free for Evaluation", and with a per-use royalty for Commercial Use, where "Free for Evaluation" means to evaluate Cogent's software and to do exploratory development and "proof of concept" prototyping of software applications, and where "Free for Evaluation" specifically excludes without limitation:

   i. use of the SOFTWARE PRODUCT in a business setting or in support of a business activity,

   ii. development of a system to be used for commercial gain, whether to be sold or to be used within a company, partnership, organization or entity that transacts commercial business,

   iii. the use of the SOFTWARE PRODUCT in a commercial business for any reason other than exploratory development and "proof of concept" prototyping, even if the SOFTWARE PRODUCT is not incorporated into an application or product to be sold,

   iv. the use of the SOFTWARE PRODUCT to enable the use of another application that was developed with the SOFTWARE PRODUCT,

   v. inclusion of the SOFTWARE PRODUCT in a collection of software, whether that collection is sold, given away, or made part of a larger collection.

   vi. inclusion of the SOFTWARE PRODUCT in another product, whether or not that other product is sold, given away, or made part of a larger product.

2. **COMMERCIAL USE:** COMMERCIAL USE is any use that is not specifically defined in this license as EVALUATION USE.

3. **GRANT OF LICENSE:** This EULA covers both COMMERCIAL and EVALUATION USE of the SOFTWARE PRODUCT. Either clause (A) or (B) of this section will apply to you, depending on your actual use of the SOFTWARE PRODUCT. If you have not purchased a license of the SOFTWARE PRODUCT from Cogent or one of Cogent's authorized resellers, then you may not use the product for COMMERCIAL USE.

   A. **GRANT OF LICENSE (EVALUATION USE):** This EULA grants you the following non-exclusive rights when used for EVALUATION purposes:

Software: You may use the SOFTWARE PRODUCT on any number of computers, either stand-alone, or on a network, so long as every use of the SOFTWARE PRODUCT is for EVALUATION USE. You may reproduce the SOFTWARE PRODUCT, but only as reasonably required to install and use it in accordance with this LICENSE or to follow your normal back-up practices.

Subject to the license expressly granted above, you obtain no right, title or interest in or to the SOFTWARE PRODUCT or related documentation, including but not limited to any copyright, patent, trade secret or other proprietary rights therein. All whole or partial copies of the SOFTWARE PRODUCT remain property of Cogent and will be considered part of the SOFTWARE PRODUCT for the purpose of this EULA.

Unless expressly permitted under this EULA or otherwise by Cogent, you will not:

i. use, reproduce, modify, adapt, translate or otherwise transmit the SOFTWARE PRODUCT or related components, in whole or in part;

ii. rent, lease, license, transfer or otherwise provide access to the SOFTWARE PRODUCT or related components;

iii. alter, remove or cover proprietary notices in or on the SOFTWARE PRODUCT, related documentation or storage media;

iv. export the SOFTWARE PRODUCT from the country in which it was provided to you by Cogent or its authorized reseller;

v. use a multi-processor version of the SOFTWARE PRODUCT in a network larger than that for which you have paid the corresponding multi-processor fees;

vi. decompile, disassemble or otherwise attempt or assist others to reverse engineer the SOFTWARE PRODUCT;

vii. circumvent, disable or otherwise render ineffective any demonstration time-outs, locks on functionality or any other restrictions on use in the SOFTWARE PRODUCT;

viii. circumvent, disable or otherwise render ineffective any license verification mechanisms used by the SOFTWARE PRODUCT;

ix. use the SOFTWARE PRODUCT in any application that is intended to create or could, in the event of malfunction or failure, cause personal injury or property damage; or

x. make use of the SOFTWARE PRODUCT for commercial gain, whether directly, indirectly or incidentally.

B. **GRANT OF LICENSE (COMMERCIAL USE):** This EULA grants you the following non-exclusive rights when used for COMMERCIAL purposes:

Software: You may use the SOFTWARE PRODUCT on one computer, or if the SOFTWARE PRODUCT is a multi-processor version - on one node of a network, either: (i) as a development systems for the purpose of creating value-added software applications in accordance with related Cogent documentation; or (ii) as a single run-time copy for use as an integral part of such an application. This includes reproduction and configuration of the SOFTWARE PRODUCT, but only as reasonably required to install and use it in association with your licensed processor or to follow your normal back-up practices.

Storage/Network Use: You may also store or install a copy of the SOFTWARE PRODUCT on one computer to allow your other computers to use the SOFTWARE PRODUCT over an internal network, and distribute the SOFTWARE PRODUCT to your other computers over an internal network. However, you must acquire and dedicate a license for the SOFTWARE PRODUCT for each computer on which the SOFTWARE PRODUCT is used or to which it is distributed. A license for the SOFTWARE PRODUCT may not be shared or used concurrently on different computers.

Subject to the license expressly granted above, you obtain no right, title or interest in or to the SOFTWARE PRODUCT or related documentation, including but not limited to any copyright, patent, trade secret or other proprietary rights therein. All whole or partial copies of the SOFTWARE PRODUCT remain property of Cogent and will be considered part of the SOFTWARE PRODUCT for the purpose of this EULA.

Unless expressly permitted under this EULA or otherwise by Cogent, you will not:

i. use, reproduce, modify, adapt, translate or otherwise transmit the SOFTWARE PRODUCT or related components, in whole or in part;

ii. rent, lease, license, transfer or otherwise provide access to the SOFTWARE PRODUCT or related components;
iii. alter, remove or cover proprietary notices in or on the SOFTWARE PRODUCT, related documentation or storage media;
iv. export the SOFTWARE PRODUCT from the country in which it was provided to you by Cogent or its authorized reseller;
v. use a multi-processor version of the SOFTWARE PRODUCT in a network larger than that for which you have paid the corresponding multi-processor fees;
vi. decompile, disassemble or otherwise attempt or assist others to reverse engineer the SOFTWARE PRODUCT;
vii. circumvent, disable or otherwise render ineffective any demonstration time-outs, locks on functionality or any other restrictions on use in the SOFTWARE PRODUCT;
viii. circumvent, disable or otherwise render ineffective any license verification mechanisms used by the SOFTWARE PRODUCT, or
ix. use the SOFTWARE PRODUCT in any application that is intended to create or could, in the event of malfunction or failure, cause personal injury or property damage.

4. **WARRANTY:** Cogent cannot warrant that the SOFTWARE PRODUCT will function in accordance with related documentation in every combination of hardware platform, software environment and SOFTWARE PRODUCT configuration. You acknowledge that software bugs are likely to be identified when the SOFTWARE PRODUCT is used in your particular application. You therefore accept the responsibility of satisfying yourself that the SOFTWARE PRODUCT is suitable for your intended use. This includes conducting exhaustive testing of your application prior to its initial release and prior to the release of any related hardware or software modifications or enhancements.

   Subject to documentation errors, Cogent warrants to you for a period of ninety (90) days from acceptance of this EULA (as provided above) that the SOFTWARE PRODUCT as delivered by Cogent is capable of performing the functions described in related Cogent user documentation when used on appropriate hardware. Cogent also warrants that any enclosed disk(s) will be free from defects in material and workmanship under normal use for a period of ninety (90) days from acceptance of this EULA. Cogent is not responsible for disk defects that result from accident or abuse. Your sole remedy for any breach of warranty will be either: i) terminate this EULA and receive a refund of any amount paid to Cogent for the SOFTWARE PRODUCT, or ii) to receive a replacement disk.

5. **LIMITATIONS:** Except as expressly warranted above, the SOFTWARE PRODUCT, any related documentation and disks are provided "as is" without other warranties or conditions of any kind, including but not limited to implied warranties of merchantability, fitness for a particular purpose and non-infringement. You assume the entire risk as to the results and performance of the SOFTWARE PRODUCT. Nothing stated in this EULA will imply that the operation of the SOFTWARE PRODUCT will be uninterrupted or error free or that any errors will be corrected. Other written or oral statements by Cogent, its representatives or others do not constitute warranties or conditions of Cogent.

   In no event will Cogent (or its officers, employees, suppliers, distributors, or licensors: collectively "Its Representatives") be liable to you for any indirect, incidental, special or consequential damages whatsoever, including but not limited to loss of revenue, lost or damaged data or other commercial or economic loss, arising out of any breach of this EULA, any use or inability to use the SOFTWARE PRODUCT or any claim made by a third party, even if Cogent (or Its Representatives) have been advised of the possibility of such damage or claim. In no event will the aggregate liability of Cogent (or that of Its Representatives) for any damages or claim, whether in contract, tort or otherwise, exceed the amount paid by you for the SOFTWARE PRODUCT.

   These limitations shall apply whether or not the alleged breach or default is a breach of a fundamental condition or term, or a fundamental breach. Some jurisdictions do not allow the exclusion or limitation of liability for consequential or incidental damages, or certain limitations of implied warranties. Therefore the above limitation may not apply to you.

6. **DESCRIPTION OF OTHER RIGHTS AND LIMITATIONS:**

   Separation of Components. The SOFTWARE PRODUCT is licensed as a single product. Its component parts may not be separated for use on more than one computer.

Termination. Without prejudice to any other rights, Cogent may terminate this EULA if you fail to comply with the terms and conditions of this EULA. In such an event, you must destroy all copies of the SOFTWARE PRODUCT and all of its component parts.

7. **UPGRADES:** If the SOFTWARE PRODUCT is an upgrade from another product, whether from Cogent or another supplier, you may use or transfer the SOFTWARE PRODUCT only in conjunction with that upgrade product, unless you destroy the upgraded product. If the SOFTWARE PRODUCT is an upgrade of a Cogent product, you now may use that upgraded product only in accordance with this EULA. If the SOFTWARE PRODUCT is an upgrade of a component of a package of software programs which you licensed as a single product, the SOFTWARE PRODUCT may be used and transferred only as part of that single product package and may not be separated for use on more than one computer.
8. **COPYRIGHT:** All title and copyrights in and to the SOFTWARE PRODUCT (including but not limited to any images, photographs, animations, video, audio, music, text and ’applets", incorporated into the SOFTWARE PRODUCT), any accompanying printed material, and any copies of the SOFTWARE PRODUCT, are owned by Cogent or its suppliers. You may not copy the printed materials accompanying the SOFTWARE PRODUCT. All rights not specifically granted under this EULA are reserved by Cogent.
9. **PRODUCT SUPPORT:** Cogent has no obligation under this EULA to provide maintenance, support or training.
10. **RESTRICTED RIGHTS:** Use, duplication, or disclosure by the U.S. Government is subject to restrictions as provided in DFARS 227.7202-1(a) and 227.7202-3(a) (1995), DFARS 252.227-7013(c)(1)(ii) (OCT 1988), FAR 12.212(a)(1995), FAR 52.227-19, or FAR 52.227-14 (ALT III), as appropriate. Manufacturer is Cogent Real-Time Systems Inc. 162 Guelph Street, Suite 253, Georgetown, Ontario, L7G 5X7, Canada.
11. **GOVERNING LAW:** This Software License Agreement is governed by the laws of the Province of Ontario, Canada. You irrevocably attorn to the jurisdiction of the courts of the Province of Ontario and agree to commence any litigation that may arise hereunder in the courts located in the Judicial District of Peel, Province of Ontario.

Table of Contents

List of Tables

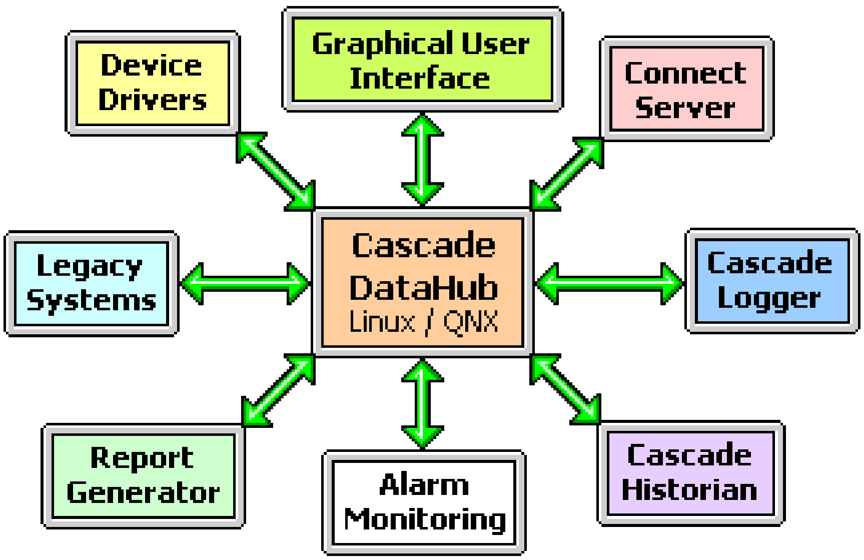
List of Figures

Chapter 1. Introduction

1.1. What is the Cascade DataHub?

- A real-time data collection and distribution program for modular QNX and Linux applications.
- RAM-resident and extremely fast.
- Needs no pre-configuration.

The Cascade DataHub is a real-time database used in process control and other modular real-time applications. It is RAM resident, and unlike other types of databases it requires no configuration.



The Cascade DataHub allows you to:

- share data among any number of programs
- eliminate communication deadlocks among cooperating programs

The Cascade DataHub forms the central data handling mechanism for many real-time and embedded systems developed in QNX and Linux. It allows you to develop applications where many different modules communicate through a common, non-blocking mechanism. The Cascade DataHub offers both a publish/subscribe model for low-latency data updates, as well as a more traditional read/write model for applications that wish to control the rate and timing of data delivery. The Cascade DataHub uses a combination of data packaging and intelligent queueing to ensure that the behaviour and communication bandwidth of one program will not adversely affect any others.

1.2. A note about the Cascade DataHub API

Cogent provides a free application programming interface (API) for the Cascade DataHub, documented in the Cogent C API manual. This API consists of a C code library and documented examples that allow you to integrate the DataHub into your own applications. The API allows you to read, write and have your programs register for exceptions with the DataHub. The example code to do this is provided at the end of the Cogent C API manual.

The API is available for download from the Cogent Web Site (http://www.cogent.ca). If you have problems downloading the API from our web site, please contact Cogent and we can arrange to have the API sent to you on diskette.

1.3. System Requirements

QNX 6

- QNX 6.1.0 or later.

QNX 4

- QNX 4.23A or later.

Linux

- Linux 2.4 or later.
- The SRR IPC kernel module, which includes a synchronous message passing library modeled on the QNX 4 send/receive/reply message-passing API. This module installs automatically, but requires a C compiler for the installation. You can get more information and/or download this module at the Cogent Web Site.

1.4. Download and Installation

You can download the Cascade DataHub from the Cogent Web Site, and then follow these instructions for installing it on your system.

Cogent software comes packaged in self-installing archives available for download, or on diskette for commerically-licensed packages. Each software package name, which we refer to in these instructions as `software_package_name`, contains the product name, version number, operating system and sometimes other information, and will end with either `.sh.gz` or `.qpr`. For example, `gamma-4.0-bin-48-Linux.sh.gz` or `CascDataHub-4.0-bld10-x86-Cogent.qpr` are typical package names. The installation procedure is standardized across Cogent products, but depends on the operating system.

1.4.1. QNX 4

Option A: Install the archive from diskette.

1. Log in as root.
2. Insert the program diskette into your QNX 4 computer.
3. Type the command: **`install`**

   and respond to the system prompts.

Option B: Install the archive from a download or received as an e-file.

1. Download or copy the `software_package_name.sh.gz` file onto your QNX 4 computer.
2. Log in as root.
3. Type the command: **`gunzip software_package_name.sh.gz`**

   This unzips the software package, and removes the `.gz` extension from the end of the filename.

4. Type the command: **`sh`** ***`software_package_name`*****`.sh`**

   and respond to the system prompts.

   If you get an error trying to install the `.sh` archive in QNX, please read the Installing program archives in QNX section of the Glossary, FAQ and Troubleshooting for help.

1.4.2. QNX 6

Option A: Use the QNX 6 Installer program. The Cogent repository is located at http://developers.cogentrts.com/repository.

Option B: Download the *`software_package_name`*`.qpr` file using the QNX 6 Voyager browser. The archive will install automatically.

Option C: Download or copy from diskette the *`software_package_name`*`.qpr` file onto your QNX 6 computer. Then (as root) run the command:

```
qnxinstall software_package_name.qpr
```

and respond to the system prompts.

1.4.3. Linux

First make sure the SRR kernel module is installed. If not, it is downloadable from the SRR for Linux page of the Cogent web site. Then follow these instructions to install the software package:

1. Download or copy from diskette the *`software_package_name`*`.sh.gz` file onto your Linux computer.
2. Log in as root.
3. Type the command: **`gunzip`** ***`software_package_name`*****`.sh.gz`**

   This unzips the software package, and removes the `.gz` extension from the end of the filename.
4. Type the command: **`sh`** ***`software_package_name`*****`.sh`**

   and respond to the system prompts.

1.4.4. Installed file locations

On whichever OS the software is installed, all files will be written to the `/usr/cogent/` directory. Depending on which packages are installed, the following subdirectories will contain the types of files shown:

`bin/`	Binary executables.
`dll/`	Dynamically-linked libraries.
`docs/`	Miscellaneous documentation. (Regular documentation is downloaded separately.)
`include/`	Header files.
`lib/`	Cogent library files.
`license`	The license file (see below).
`require/`	Lisp or Gamma files used by Gamma or its extensions.
`src/`	The source code for examples, tests, tutorials, etc.

1.4.5. Installing licenses

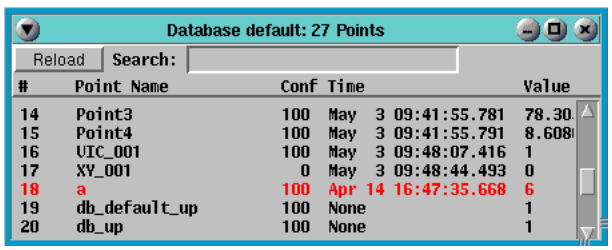
Licenses to use the software can be purchased from Cogent. To install a license, you need to copy the license string into the `/usr/cogent/license` file. If this file does not exist on your system, just create one as a text file and list the license strings, one per line.

1.5. Quick Start

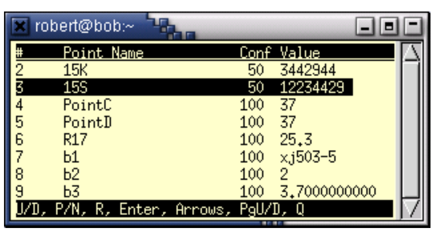
Once the Cascade DataHub is installed, you can run it from a terminal. From a second terminal you can run the DataHub Viewer in console mode to get a window into the Cascade DataHub. And from a third terminal, you can verify the data using the **waiter** command. Try it. Open three terminals and issue the following commands:

Terminal	Command	What it does
1	`[sh]$` **`datahub`**	Starts the DataHub.
2	`[sh]$` **`dhview`**	Shows the contents of the Cascade DataHub.
3	`[sh]$` **`waiter`**	Lets you verify the status of the data.
1	`[sh]$` **`writept test 25`**	Sends test data.

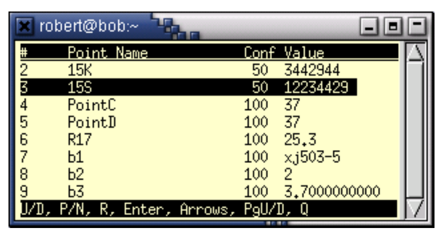
Here is what you should see on the three terminals:



Terminal 1



Terminal 2



Terminal 3

You can now use the **writept** command to write new values to the `test` point, like this:

```
[sh]$ writept test 7975
```

Or, create new points and write values (numbers or strings) to them. For example:

```
[sh]$ writept newpt "Hello world"
```

This is the basic concept of the Cascade DataHub. It receives and transmits data from and to any process that is registered with it. Chapter 2, *Using the Cascade DataHub* gives more detailed information.

1.6. Cogent Product Integration

Cogent products work together to support real-time data connectivity in Windows, Linux, and QNX. They can be dynamically integrated as a group of modules where each module connects to any other module(s) as needed. New modules can be added and existing modules reconfigured or modified, all during run-time. Data in any module of the system can be collected and redistributed to any other module via the Cascade DataHub and Cascade Connect. Communication with field devices is provided by one of several Cogent Device Drivers. Historical records of unlimited size can be maintained and queried with the Cascade Historian, and ASCII text files can be logged with the Cascade TextLogger.

Custom programs written in C or C++ can interface with the system, using the Cogent C API or the DataHub APIs for C++, Java, and .NET. In addition, Cogent's own dynamically-typed object-oriented programming language, Gamma, is fully compatible with all modules. User interfaces can be created in Gamma, which supports Photon in QNX and GTK in Linux.

1.7. Where can I get help?

If you are having problems with a Cogent product, first check the Troubleshooting Guide. If you can't find the answer there, you can contact Cogent Real-Time Systems, Inc. for technical support for any product you have purchased.

- Email: <`support@cogent.ca`>
- Phone: 1-888-628-2028
- Fax: (905) 702-7850

Chapter 2. Using the Cascade DataHub

2.1. Starting up and shutting down.

Follow these steps to start the Cascade DataHub.

1. Ensure that `/usr/cogent/bin` is in your `PATH` (check this using the **set** command).
2. Start the Cascade DataHub using the **datahub** command with the appropriate arguments, (no ampersand '&' is required). For example:

   ```
   [sh]$  datahub
   ```

   The example above would start the DataHub in the default domain (called "`default`"). Another example is:

   ```
   [sh]$  datahub -d test
   ```

   This second example starts the DataHub in an application domain called "`test`".

Once started you can check that the DataHub is running by using the **nsnames** command. For example:

```
[sh]$  nsnames
Name          Domain     Queue          NID    PID
/dh/test      test       /dh/test       0      15367
```

The DataHub derives its name and queue name from the domain(s) that it is operating in. If no domain is specified, the DataHub takes the name `/dh/default`. Please refer to Section 2.4.3, *Domains and Names* for more information on this. The NID and PID will vary depending on circumstances.

2.2. Testing the installation

1. In a shell, type the command:

   ```
   [sh]$  writept test1 25
   ```

   This will create the point `test1` in the DataHub and assign it the value `25`. **writept** does not return any value.

   You will need to use this syntax:

   ```
   [sh]$  writept test:test1 25
   ```

   if you started the DataHub with the command `datahub -d test`. This is because the point `test1` is in the `test` domain. Please refer to Section 2.4.3, *Domains and Names* for more information.

2. At the shell, type the command:

   ```
   [sh]$  readpt test1
   ```

   or

   ```
   [sh]$  readpt test:test1
   ```

   The results should look like this:

   ```
   Point: test1
   Value: 25
   Time:  Jun 14 10:43:38.896
   Conf: 100
   Lock: 0
   Secur: 0
   ```

These tests establish that you can read and write data points to the Cascade DataHub. The syntax for **writept** and **readpt** are described in the Utilities reference, and their source code is is normally installed in the `/usr/cogent/src/datahub` directory.

Shutting down

The simplest way to shut down the DataHub is to send a **kill -9** command for the DataHub's process ID. A more graceful and orderly method is to send the `exit` command using the **lsend** utility, like this:

```
[sh]$  ps -aux
...
name   19350  0.0  0.3  3160 1612 ?        S    16:37   0:00 datahub
name   19354  0.0  0.1  2856  868 pts/0    R    16:39   0:00 ps -aux
[sh]$ lsend 19350
/dh/19350> exit()
  </dh/19350 no longer reachable>
/dh/19350(disconnected)>Ctrl-C
[sh]$
```

If the DataHub shuts down in any way other than using **lsend**, it takes Linux or QNX a couple of minutes, more or less, to free up the TCP socket that the DataHub was using. So if you kill the DataHub, you'll have to wait a couple of minutes before attempting to restart it.

2.3. Configuration

The Cascade DataHub can be configured with a configuration file at startup, or dynamically during runtime. The commands used for configuration are documented in the reference, in the Cogent DataHub manual. The most commonly-used commands are:

tells the Cascade DataHub that the client is running.
identifies the client domain name.
enables or disables DDE client capabilities.
enables or disables DDE server capabilities.
enables or disables mirror master capabilities.
enables or disables mirror slave capabilities.
enables or disables TCP server capabilities.
establishes a heartbeat message.
ignore a given point.
registers the Cascade DataHub on a given domain.
requests the value of a point.
gets current data when client connection is made.
suspends data flow.

2.3.1. Using a Configuration File at Startup

The Cascade DataHub can read a configuration file at startup, using the -f option. The commands in the configuration file are written using Lisp syntax. This consists of the name of the command, followed by a space-separated list of arguments, all enclosed in parentheses, like this:

```
(command arg1 arg2 arg3 ...)
```

The commands are commonly written one per line. Comments are denoted by a semicolon (`;`) at the beginning of each comment line. For example:

```
; This line is a comment.
(tcp_service 4601)
(enable_tcp_server 1)
(enable_mirror_master 1)
```

2.3.2. Dynamic Run-Time Configuration

There are several ways that the Cascade DataHub can be configured or reconfigured dynamically during run time:

- **Using the lsend or gsend command-line tools.** These two commands are similar, except one uses Lisp syntax, as explained above, while the other uses Gamma syntax. Please refer to the **lsend** documentation in the Cogent C API manual for more information.

  Here is an example using **lsend**:

  1. Start the DataHub and declare the `example` domain.

     ```
     [sh]$  datahub -d example
     ```

  2. Start **lsend** using the name of the DataHub. This consists of the string `/dh/` followed by the domain name. If no domain is declared, the DataHub takes its name from its PID. Please refer to Section 2.4.3, *Domains and Names* for more information.

     ```
     [sh]$  lsend "/dh/example"
     /dh/example>
     ```

     When **lsend** starts it gives you a prompt with the name of the receiving program in it.

  3. At the prompt generated by **lsend**, send commands using the Lisp syntax explained above:

     ```
     /dh/example> (alive)
     (success "alive")
     /dh/example> (heartbeat 200)
     (success "heartbeat" "200")
     /dh/example>
     ```

     For each command, **lsend** displays the return value from the DataHub in Lisp syntax.

  Here is the same example using **gsend**:

  ```
  [sh]$  datahub -d example
  [sh]$  gsend "/dh/example"
  /dh/example> alive;
  (success "alive")
  /dh/example> heartbeat (200);
  (success "heartbeat" "200")
  /dh/example>
  ```

  The return values for **gsend** from the DataHub are in also in Lisp syntax.

- **Sending a command from a C program.** The Cogent C API has functions that allow you to send configuration information to the Cascade DataHub. These are explained in the Communicating with the Cascade DataHub section of the Cogent C API manual.
- **Using the Gamma `send` and `send_async` functions.** Cogent's Gamma programming language uses these two functions to send commands to the Cascade DataHub and other Cogent products. Please refer to `send` and `send_async` in the Gamma manual reference for more details.

2.4. Working with Data

2.4.1. Data Points

Each value stored in the Cascade DataHub is called a *point*. A point has the following attributes:

- **Name:** a character string. Currently the only limit on length is internal buffer size, about 1000 bytes by default.

- **Value:** an integer, floating-point number, or character string.
- **Time:** the date and time of the last significant change to the point's value, confidence, quality or other status information.
- **Quality:** the quality of the connection, assigned by the DataHub for this point, such as `Good`, `Bad`, `Last known`, `Local override`, etc. The possible values are those supported by OPC in Microsoft Windows.
- **Confidence:** a value from `0` to `100` that indicates as a percentage the probability that the value shown for the point is actually its true value. This feature can be accessed and changed only by using the API. The DataHub never uses confidence itself, but carries it for use by client applications.

The Cascade DataHub does not require you to configure the names or types of data points you will be using in your system. If your program writes a data value to the DataHub and the point does not exist, the DataHub will create the point. If a program registers for exceptions for named points that currently do not exist in the DataHub then the DataHub will create those points and register the client program at the same time. If a program tries to read the value of a point that doesn't exist in the DataHub, then the DataHub will create the point and return a default zero value with a zero confidence to the client program.

The DataHub does not limit the size of a point data message. The only limits are those imposed by the operating system. This limit is 64000 bytes in QNX using SRR, 128000 bytes in Linux using the SRRIPC Module, and unlimited for TCP. Bear in mind that very large values will take more time to be transmitted over a network.

Typically, the Cascade DataHub is started before other application modules. Then, other tasks are started that communicate with the DataHub (to request exceptions, send data, or both). Memory (RAM) is allocated for the points as they are created by the DataHub.

It is not possible to directly delete points from the DataHub. Should a point no longer be in use by any participating program, when the DataHub is shut down and restarted, the point will no longer appear.

2.4.2. Registering for Exceptions

The normal way to receive data is to have your program *register for an exception* on a data point. Once you have registered for an exception, the DataHub will send you an update whenever the value for that point changes.

The **waiter** utility registers for exceptions with a DataHub and displays any new point values that it receives. The source code for this example will help you develop applications that effectively utilize the DataHub. See **waiter** for details about the syntax for **waiter**, and its source code is normally installed in the `/usr/cogent/src/datahub` directory.

2.4.3. Domains and Names

The Cascade DataHub divides data into *domains*. This provides namespace separation and avoid conflicts when working with multiple third-party data sources. In addition, using domains for a large network can significantly reduce network traffic. All references for point values that reside in a domain other than `default` are referenced using a domain prefix of the following format: `domain:pointname`

The Cascade NameServer assigns the DataHub a name for each domain. These names are used for communication with other programs, and can be viewed using the **nsnames** command. For example:

```
[sh]$  datahub -d test1 -d test2
[sh]$  nsnames
Name      Domain Queue      NID PID
/dh/test1 test1  /dh/test1  0   13446
/dh/test2 test2  /dh/test1  0   13446
```

In some cases the DataHub will only register its name with a process ID on it. This is useful if you don't specify a domain with the `-d` option or in the configuration file, and the DataHub is acting as a TCP master (see below). For example:

```
[sh]$ datahub
[sh]$ nsnames
Name        Domain  Queue      NID PID
/dh/13440   default /dh/13440  0   13440
/dh/default default /dh/13440  0   13440
```

As clients ask for points in various domains, the DataHub will register the appropriate names with **nserve**, if possible. If you have more than one DataHub serving the same domain name, only the first one will be allowed to register the `/dh/`*`domain`* name with nserve.

2.4.4. Assemblies, Subassemblies, Attributes, and Properties

Within a domain, data can be arranged hierarchically as assemblies, subassemblies, attributes, and properties. Each assembly can have zero or more attributes and zero or more subassemblies, and each attribute can have zero or more properties. Subassemblies can have subassemblies. You can think of assemblies and subassemblies as branches in a tree, and attributes as the leaves. Here is an example of what a tree might look like:

```
Domain
    Assembly
        Subassembly (zero or more)
            Attribute  (zero or more)
                Property  (zero or more)
        Attribute...
        Attribute...
        Attribute...
            Property...
            Property...
            Property...
        Subassembly
            Subassembly
                Attribute...
                  Property...
                  Property...
            Attribute...
            Attribute...
                Property...
    Assembly...
```

and so on.

The written syntax for all of these levels uses a dot (`.`) to divide the names, rather than a colon that was used for the domain name. Hence, the syntax of point in a property in an attribute in a subassembly in an assembly in a domain would be:

```
domain:assembly.subassembly.attribute.property
```

Properties describe the attributes in more detail. An attribute can have a *default property* such that if you interact with the attribute point directly you will in fact be interacting with its default property. For example, an item might be `plant.temperature`, with properties `value`, `highlimit`, `units`. This would create 4 tags:

```
plant.temperature
plant.temperature.highlimit
plant.temperature.units
plant.temperature.value
```

The tags:

```
plant.temperature
plant.temperature.value
```

are aliases of one another. Both refer to the default property of `plant.temperature`. If you specify no property at all for an item, the item takes on the default property.

2.4.5. Attributes and Types

It is common for attributes to contain the same type of information. For example, all temperatures in a system are likely to share units, high alarm level, and value. To avoid repeating this information for each and every temperature in the sytem, we use a *type*. A type is the prototype, or class, of an attribute. You define a type and its properties first, and then define attributes of that type on assemblies. When the assembly is instantiated, its attributes are instantiated by creating an attribute and then assigning the properties to it that are associated with the attribute's type.

There is an alternative to using types and attributes as described here, a *private attribute*. A private attribute provides a one-command means of creating an attribute on an assembly without having to define a type. However, this means that the attribute properties cannot be shared across more than one attribute in the assembly or in other assemblies. This is normally only used internally with machine-generated hierarchies. In most cases it is better to use types and attributes.

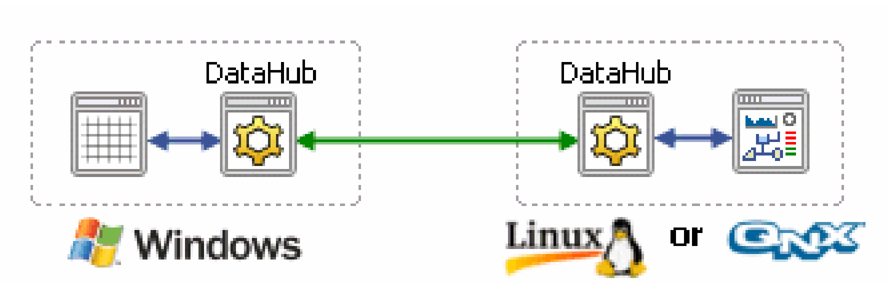
2.5. Mirroring Data to Windows or other nodes in Linux or QNX

The Cascade DataHub can *mirror* data with one or more other DataHubs in running in Windows, as well as Linux or QNX, across a LAN, WAN, or the Internet, using TCP. Mirroring means that the data and any updates to that data on one DataHub are exactly mirrored across the network onto another DataHub, and vice-versa. The only difference between the master DataHub and slave DataHub is that the slave initiates the connection, and a connection license is consumed on the master. Once the connection is established, they function exactly the same.

Each participating DataHub must be configured as mirroring master (server) or mirroring slave (client), or in some cases, both. If your system requires it, a single DataHub can act as a master to one DataHub and as a slave to another.

Each master-slave pair must have matching port numbers and domain names. Specifying port numbers and domain names is explained below.

2.5.1. Exchanging data between Windows and Linux/QNX

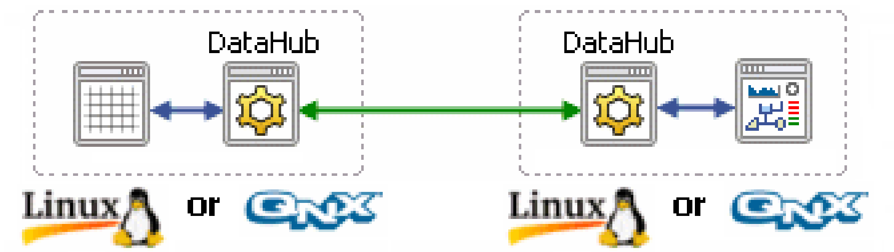


There are three possible ways to set this up:

1. To have the Windows DataHub initiate the connection, you would need to set it up as a slave, and set up the Linux or QNX DataHub as a master.
2. To have the Linux or QNX DataHub initiate the connection, you would need to set it up as a slave, and set up the Windows DataHub as a master.

3. As a third alternative, you can use Cascade Connect instead of the DataHub in Windows, however you must keep in mind that Cascade Connect always functions as a slave, and it only connects to DDE-enabled programs. Please refer to the Cascade Connect manual for details on setting it up for mirroring. You would set it up the Linux or QNX DataHub as a master

2.5.2. Exchanging data between Linux/QNX and Linux/QNX



For this scenario, you would set up whichever DataHub you wanted to initiate the connection as a master, and set up the other DataHub as a slave.

2.5.3. Mirroring Master Setup - Linux or QNX

You can set up the Cascade DataHub to act as a mirroring master on Linux or QNX in either of these two ways:

1. **Run the DataHub with the -p option.** This tells the Cascade DataHub to listen as a TCP master on the port or service you specify. Normally you would use port `4600`, which we have arbitrarily chosen as the "normal" port, but any port number will do so long as the client (slave) uses the same port number.

   ```
   [sh]$  datahub -p port/service
   ```

   For example:

   ```
   [sh]$  datahub -p 4600
   ```

2. **Create a configuration file or use an existing one.** Make sure the following lines are in the file:

   ```
   (tcp_service 4600)
   (enable_tcp_server 1)
   (enable_mirror_master 1)
   ```

   Then run the DataHub with the `-f` option:

   ```
   [sh]$  datahub -f /path/to/my/configfile.cfg
   ```

   For example:

   ```
   [sh]$  datahub -f /usr/local/misc/dhconfig.cfg
   ```

2.5.4. Mirroring Slave Setup - Linux or QNX

You can set up the Cascade DataHub to act as a mirroring slave on Linux or QNX in either of these two ways:

1. **Run the DataHub with the -m, -M and -n options.**

   ```
   [sh]$  datahub -m port -M address -n domain
   ```

For example:

```
[sh]$  datahub -m 4600 -M 192.168.3.15 -n test
```

2. **Create a configuration file or use an existing one.** Make sure the following lines are in the file:

```
(create_domain domain_M)
(mirror_master address port domain_S domain_M)
(enable_mirror_slave 1)
```

The **create_domain** command is optional. It is used in this example to show how to make a domain on the slave DataHub that matches a domain on the master DataHub. The **mirror_master** command then sets up mirroring from *domain*$_M$ on the master to a different domain, *domain*$_S$, on the slave. You can also mirror domains with the same name, say `default`, by simply specifying the same domain name for master and slave:

```
(mirror_master 192.168.3.15 4600 default default)
```

Once the file is ready, run the DataHub with the `-f` option:

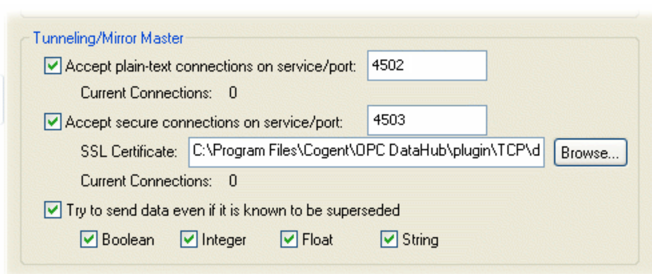
```
[sh]$  datahub -f /path/to/my/configfile.cfg
```

For example:

```
[sh]$  datahub -f /usr/local/misc/dhconfig.cfg
```

2.5.5. Mirroring (Tunnel) Master Setup - Windows

You can configure your DataHub to act as a master for either plain-text tunneling, secure tunnelling using SSL, or both. Each mode uses a separate port number or service name.



If you enter a name for the service/port instead of a number, that name must be listed in the Windows `services` file. Please refer to The Windows Services file Appendix for details.

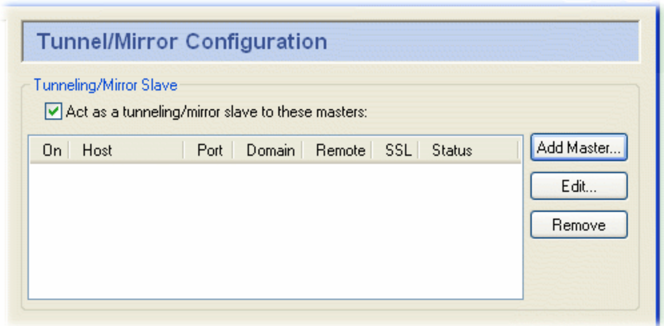
The DataHub installs an SSL Certificate for you. If you wish to move it or use a different one, you can change the directory path here. The SSL implementation uses the default SSL-3 encryption cipher: DHE-RSA-AES256-SHA. This is a 256-bit encryption. The server and client negotiate the best encryption based on what both can support. The DataHub does not validate the SSL certificate with any outside certificate authority. It uses the SSL connection for encryption only, not authentication.

You can also configure the master to attempt to send "old" data (superseded by more recent data). Check any or all of Boolean, Integer, Float, or String that apply to the kind of superseded data that you wish to have sent.

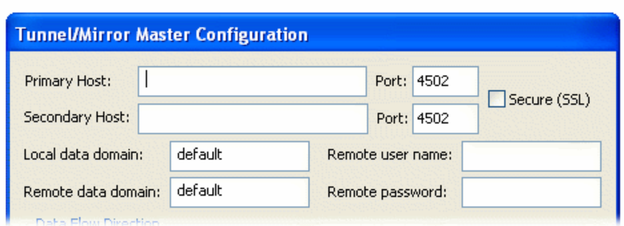
To optimize throughput using this option, please refer to .

2.5.6. Mirroring (Tunnel) Slave Setup - Windows

Check the Act as a tunneling/mirror slave to these masters box to have the Cogent DataHub act as a slave.



To add a master for this mode, click the Add Master... button. To edit a master, double-click it, or select it and press the Edit... button. Either button opens the Tunnel/Mirror Master window:



Type in the following information:

Primary/Secondary Host

> The name or IP address of the host computer. This slave DataHub will alternate attempts to connect first on the primary host, then on the secondary host, back and forth until a connection is made. The secondary host is optional, and if not entered, all attempts to reconnect will be on the primary host. If the connection is interrupted, the DataHub will again alternate attempts at reconnection on the primary and secondary hosts.

Port

> The port number or service name as entered in the Master service/port entry box of the master on the remote computer.

Secure (SSL)

> You can establish a secure connection using SSL tunneling as long as the tunnelling master DataHub you are attempting to connect to has been configured for secure connections. (See above.)

Local data domain

> The local Cogent DataHub data domain for this slave. It is common, but not necessary, to create or use an existing local data domain that has the same name as the remote data domain.

Remote data domain

> The name of the remote Cascade DataHub data domain, which is the tunneling master. Point names will be mapped from that data domain into the local data domain, and vice versa.

Remote user name

The user name for TCP security, established on the tunneling master, using the DataHub Security option in the Properties window.

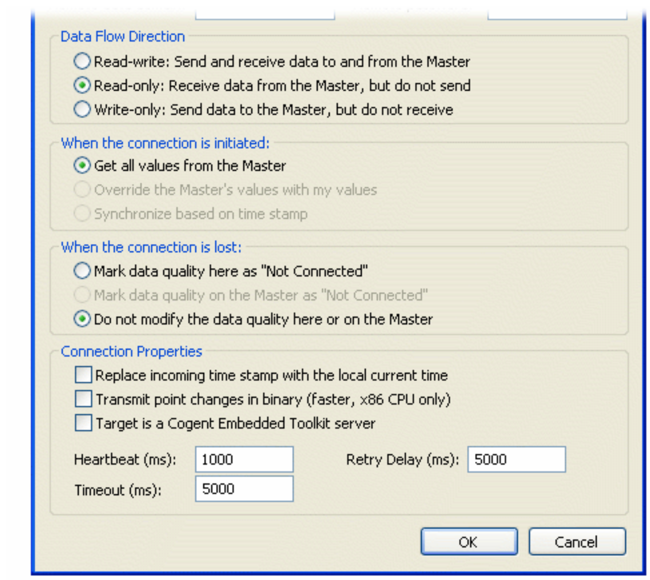
Remote passwork

The password for TCP security, established on the tunneling master, using the DataHub Security option in the Properties window.

There is a DataHub running on Cogent's server that you can connect to for testing. Here are the parameters you will need to enter for it:

- **Primary Host:** `developers.cogentrts.com`
- Port: `4502`
- **Local data domain:** `test`
- **Remote data domain:** `test`

You now have several options for the mirrored connection.



1. **Data Flow Direction** lets you determine which way the data flows. The default is read-only data flow from master to slave, but you can set up a read-write or write-only connection by choosing those options.

   To optimize throughput, check the Read-only: Receive data from the Master, but do not send option. Only do this if you actually want a read-only connection. If you do not require read-write access, a read-only tunnel will be faster.

2. **When the connection is initiated** determines how the values from the points are assigned when the slave first connects to the master. There three possibilities: the slave gets all values from the master (the default), the slave sends all its values to the master, or the data from master and slave gets synchronized. The availability of these options depends on the data flow direction selected above.

3. **When the connection is lost** determines where to display the data quality as "Not Connected", on the master, on the slave, or neither.

   If you have configured When the connection is initiated as Synchronize based on time stamp (see above), then this option must be set to Do not modify the data quality here or on the Master to get correct data synchronization.

4. **Connection Properties** gives you these options:

- **Replace incoming timestamp...** lets you use local time on timestamps. This is useful if the source of the data either does not generate time stamps, or you do not trust the clock on the data source.
- **Transmit point changes in binary** gives users of x86 CPUs a way to speed up the data transfer rate. Selecting this option can improve maximum throughput by up to 50%, depending on the type of data being transmitted. This option uses a more efficient message encoding scheme than the default ASCII encoding, but it will only work if both sides of the tunnel are running on an x86 architecture CPU. This would be typical of Windows communicating with Linux or QNX, or with another Windows computer. Numeric data benefits most from this option.
- **Target is a Cogent Embedded Toolkit server** allows this slave to connect to an Embedded Toolkit server rather than to another DataHub.
- **Heartbeat** sends a heartbeat message to the master every number of milliseconds specified here, to verify that the connection is up. Setting this value to `0` stops the heartbeat from being transmitted.
- **Timeout** specifies the timeout period for the heartbeat. If the slave DataHub doesn't receive a response from the master within this timeout, it drops the connection. You must set the timeout time to at least twice the heartbeat time. Setting this value to `0` will cause the DataHub to rely on the TCP implementation for detecting a broken connetion. This can be useful when your network connection is very slow. Please refer to for details.
- **Retry** specifies a number of milliseconds to wait before attempting to reconnect a broken connection.

2.6. Viewing Data

You can see what's happening in the Cascade DataHub using the Cascade DataHub Viewer utility. It can operate in any of three modes: console, QNX Photon, and X Windows.

2.6.1. Console Mode

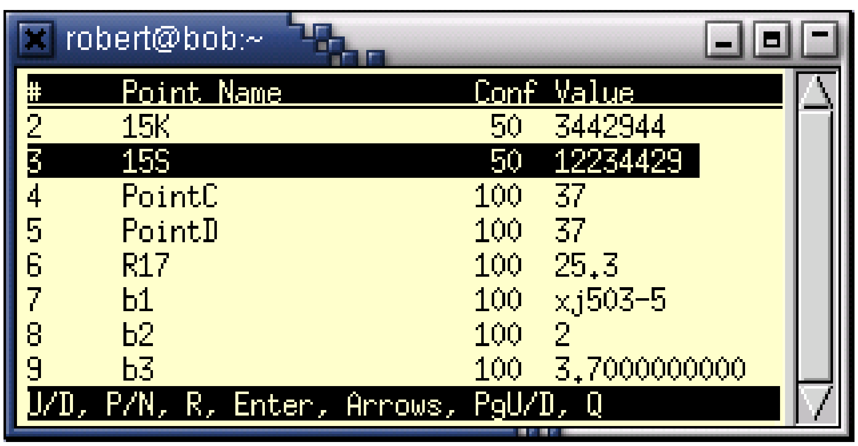


Figure 2-1. The Cascade DataHub Viewer in Console Mode

Starting:

Type **dhview** at the shell prompt to start the viewer.

Navigation:

Move the selection bar by pressing the up and down arrows.

Scroll up and down a line at a time by pressing the **U** and **D** keys; or a page at a time by pressing the **P** and **N** keys or **PgUp** and **PgDn**.

Making Changes:

You can change data point values in the DataHub from the DataHub Viewer.

New values get written as soon as the **Enter** key is pressed.

Choose a point by moving the selection bar to the desired point and pressing **Enter**. A small dialog box opens that allows you to edit the point's value, confidence, and security.

Edit a chosen point by pressing the **Backspace** key to delete the existing entry; then type in the new information. Use the **Tab** key to move between fields. Use the **Esc** key to close the dialog window, ignoring any changes and keeping the original entry. Use the **Enter** key to enter the changes and close the dialog box.

Retransmit all points from the DataHub to the viewer by pressing the **R** key. This is useful to see those points that were created but haven't been assigned a value, such as through a call to the Gamma function `read_point` or `register_point`.

Quitting:

Type **Q** to quit the Cascade DataHub Viewer.

2.6.2. QNX Photon Mode

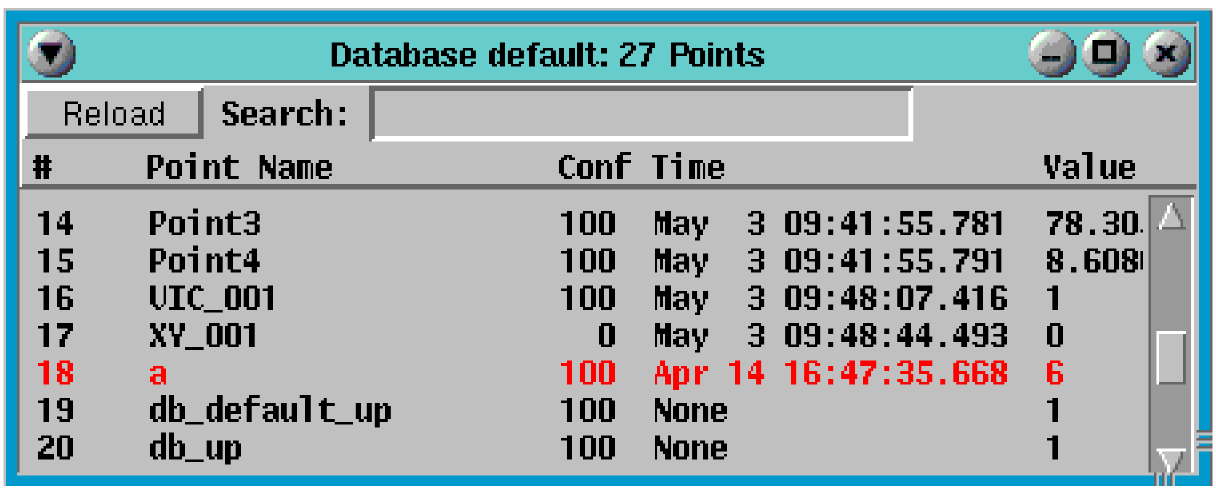


Figure 2-2. The Cascade DataHub Viewer for QNX Photon

In addition to the features in console mode, the Photon mode of the Cascade DataHub Viewer has a search feature, and its display includes a timestamp field. It also highlights in red any locked points.

Starting:

Type **`phdhview`** at the shell prompt to start the viewer.

Navigation:

Scroll up and down by using the scrollbar, the arrow keys, or **PgUp** and **PgDn**.

Find a point by typing its name in the **`Search`** entry field.

Making Changes:

You can change data point values in the DataHub from the DataHub Viewer.

! New values get written as soon as the **Apply** button is pressed.

Choose a point by clicking on it. A small dialog box opens that allows you to edit the point's value, confidence, security, and locked status.

Edit a chosen point by deleting the existing entry; then type in the new information. Use the **Tab** key or click the mouse to move between fields. Press the **Cancel** button to close the dialog window, ignoring any changes and keeping the original entry. Press the **Apply** button to enter the changes.

Retransmit all points from the DataHub to the viewer by pressing the **Reload** button. This is useful to see those points that were created but haven't been assigned a value, such as through a call to the Gamma function `read_point` or `register_point`.

Quitting:

Press the circular **X** close button in the window title bar to quit the Cascade DataHub Viewer.

2.6.3. X Windows Mode

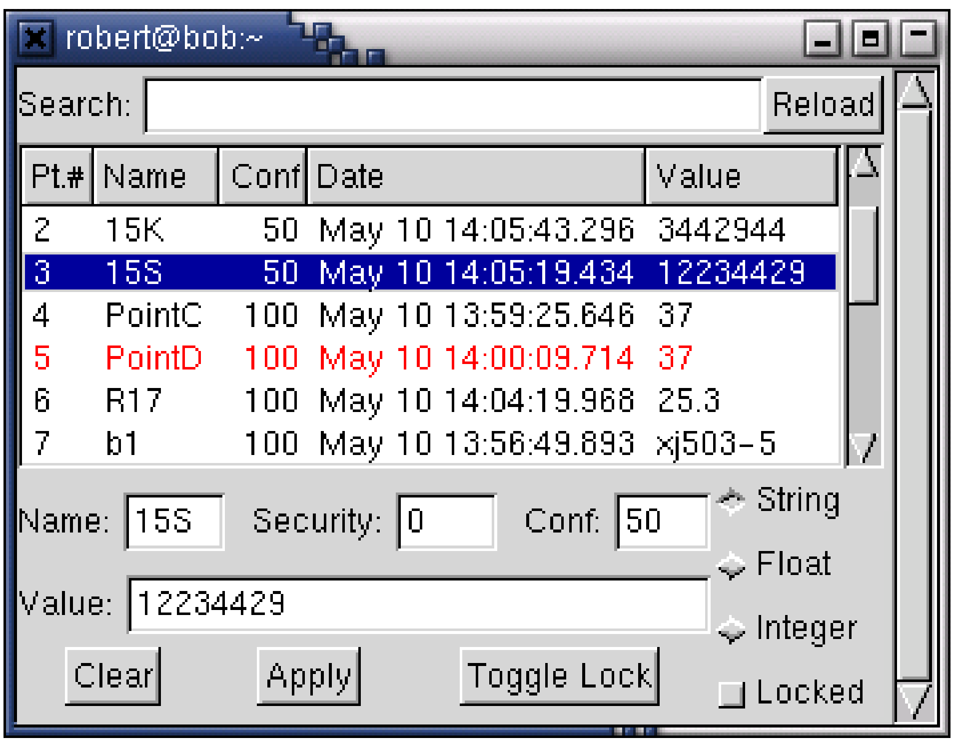


Figure 2-3. The Cascade DataHub Viewer for X Windows

In addition to the features in console mode, the X Windows mode of the Cascade DataHub Viewer has fields for changing point values, permanently displayed at the bottom of the viewer. It has a search feature, and its display includes a timestamp field for every point. It also highlights in red any locked points.

You must have GTK to use this mode.

Starting:

Type `xdhview` at the shell prompt to start the viewer.

Navigation:

Scroll up and down by using the scrollbar.

Find a point by typing its name in the **Search** entry field.

Making Changes:

You can change data point values in the DataHub from the DataHub Viewer.

New values get written as soon as the **Apply** button is pressed.

Choose a point by clicking on it. The point's information will be displayed in the data fields at the bottom of the viewer.

Edit a chosen point by deleting the existing entry; then type in the new information. Use the **Tab** key or click the mouse to move between fields. Press the **Clear** button to remove all entries, ignoring any changes and keeping the original entries. Press the **Apply** button to enter the changes.

If you change the name of a point, the Cascade DataHub Viewer will automatically create a new point in the DataHub with that name and any entered values.

Change a point type by clicking the appropriate button: **String**, **Float**, or **Integer**. Press the **Apply** button to enter the changes.

Lock or unlock a point by clicking the **Toggle Lock** or **Locked** button. Press the **Apply** button to enter the changes.

Write a new point by pressing the **Clear** button and entering the point name, confidence (usually `100`) and value. Press the **Apply** button to enter the changes.

Retransmit all points from the DataHub to the viewer by pressing the **Reload** button. This is useful to see those points that were created but haven't been assigned a value, such as through a call to the Gamma function `read_point` or `register_point`.

Quitting:

Press the square **X** close button in the window title bar to quit the Cascade DataHub Viewer.

2.7. Features

2.7.1. Exceptions and Echoes

The Cascade DataHub allows programs to register for exceptions on point value changes. When a point changes value in the DataHub, all clients that have registered an interest in that point are notified.

The Cascade DataHub not only allows its clients to register and receive exceptions on data points, but it also provides a special message type called an *echo* that is extremely important in multi-node or multi-task applications.

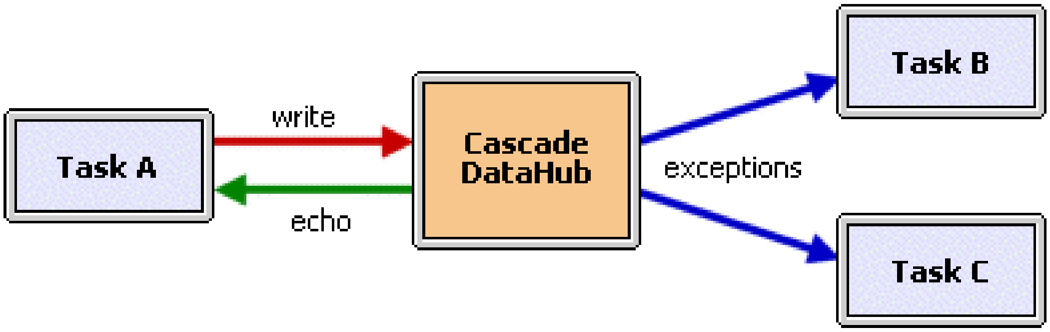


Figure 2-4. Exceptions and Echoes

When the Cascade DataHub receives a new data point it immediately informs its registered clients of the new data value. The clients will receive an asynchronous *exception* message. In some circumstances, the client that sent the new data value to the DataHub is also registered for an exception on that point. In this case, the originator of the data change will also receive an exception indicating the data change. When there are multiple clients reading and writing the same data point one client may wish to perform an action whenever another client changes the data. Thus, it must be able to differentiate between exceptions which it has originated itself, and ones which originate from other clients. The Cascade DataHub defines an *echo* as an exception being returned to the originator of the value change.

In certain circumstances, the lack of differentiation between exceptions and echoes can introduce instability into both single and multi-client systems. For example, consider an application consisting of the DataHub mirroring data to a DataHub in Windows. The Windows DataHub communicates with Wonderware's InTouch program. InTouch communicates using DDE, which does not make the distinction between exceptions and echoes. A data value delivered to InTouch will always be re-emitted to the Windows DataHub, which in turn will re-emit the value to the Linux or QNX DataHub. The Linux or QNX DataHub will generate an exception back to the Windows DataHub which will pass this exception on to InTouch. InTouch will re-emit the value, and so on. A single value change will cause an infinite communication loop. There are many other instances of this kind of behavior in asynchronous systems. By introducing the echo capability into the Cascade DataHub, the cycle is broken immediately because it recognizes that it should not re-emit a data change that it originated.

The echo facility is necessary for another reason. It is not sufficient to simply not emit the echo to the originating task. If two tasks read and write a single data point to the DataHub, then the DataHub and both tasks must still agree on the most recent value. When both tasks attempt to write the point, one gets an exception and updates its current value to agree with the DataHub and the sender. If both tasks simultaneously emit different values, then the task whose message is processed first will get an exception from the first, and the first will get an exception from the second. In effect, the two tasks will swap values, and only one will agree with the DataHub. The echo message solves this dilemma by allowing the task whose message was processed second to receive its own echo, causing it to realize that it had overwritten the exception from the other task.

2.7.2. Asynchronous Messages

Whenever multiple tasks are communicating there is a chance for a deadlock situation. The Cascade DataHub is at the centre of many mission critical applications because it provides real-time data to its clients without the threat of being blocked on the receiving task. The Cascade DataHub never blocks on a task that is busy. It is always able to receive data from clients because it uses the Cascade QueueServer (**qserve**) to handle outgoing messages.

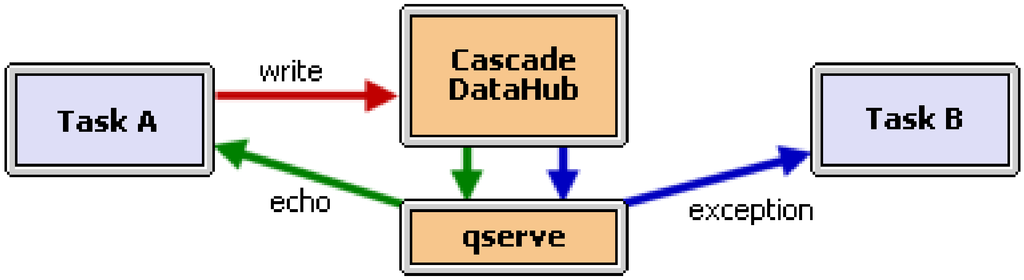


Figure 2-5. Asynchronous Messages

The DataHub only ever sends messages to **qserve** program, which is optimized so that it never enters a state where it cannot accept a message from the DataHub.

2.7.3. Network Access in QNX 4

The Cascade DataHub works across any QNX 4 network. Only the **qserve** and **nserve** tasks (approx. 100 KB RAM) need to be run on the network computer; all other tasks remain on the machine with the DataHub.

2.7.4. Confidence Factors

All data points are created with an associated confidence factor that is delivered with every point value. Any writing program may set confidence factors. This lets you change the confidence on a point value to reflect uncertainty and can be used in more advanced control strategies to 'weight' actions and responses to alarm states.

2.7.5. Security and Point Locking

The Cascade DataHub provides facilities for implementing security and point locking. It respects security levels and locked points, but the application programmer is responsible for how that security is allotted.

Changing security levels and locking points can be done through an application, or through the Cascade DataHub Viewer. For this reason, it is important to either restrict access to the Cascade DataHub Viewer, or to modify its source code to restrict access to its security features.

Generally speaking, the Cascade DataHub assigns every task a security level, expressed as an integer ranging from 0 (the default) to 32,767. Every DataHub point also has a security level, within the same range. If a task security level is greater than or equal to a point security level, then that task has full access to the point. It can register it, read it, lock it, unlock it, write to it, and change its security to any level up to and including the task's own level. On the other hand, if a task security level is less than a point security level, that task can read the point and register it, but nothing else.

The point locking feature is useful for debugging, as it allows you to prevent a function from writing to a point or group of points at the points themselves, rather than altering code.

2.7.6. Unlimited Point Count

Because the Cascade DataHub is RAM resident and requires no pre-configuration, the size of the DataHub is only limited by the available system resources.

Table 2-1. Memory Usage

Program	Size on Disk	Size in Memory
DataHub	52 KB	110 KB
Cascade QueueServer	18 KB	69 KB
Cascade NameServer	43 KB	146 KB (networked) 93 KB (standalone)
Each point requires approximately 100 bytes, which includes memory allocation overhead.		

2.7.7. Cascade DataHub performance

Most of the CPU time used by the demo is consumed updating the screens. The Cascade DataHub, by itself, has a throughput of about 2500 points/sec on a Pentium 133. This number is based on using single-point messages rather than using efficient packing techniques, which would increase throughput. Also, bear in mind that a DataHub message includes transmission through the asynchronous queue server.

The lookup of points is done in logarithmic time (i.e. the time does not grow linearly with the number of points in system).

Chapter 3. Data Transmission

In this section we discuss methods of data transmission as they apply to the Cascade DataHub.

3.1. Synchronous data transmission

Consider the following diagram that represents two programs (or tasks) communicating with each other using send/receive/reply message protocol.

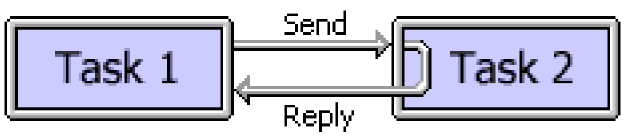


Figure 3-1. Synchronous data transmission

Task 1 sends a message to task 2. Using synchronous transmission, task 1 will not be able to continue processing until it has either received a reply from task 2 or the message transmission has failed. The message may fail to send due to a couple of reasons. The recipient task may die which means the message will fail to be sent, or the attempt to send the message may time-out waiting for a reply.

There may be a substantial time delay associated with task 2 processing the message and then sending back the reply to task 1. In the example above, task 1 is said to be 'blocked' on task 2 until it receives the reply to its original message.

In many mission critical applications, having one task blocked on another task is unacceptable because the blocked tasks are unable to perform other duties for the duration of the message transaction.

Synchronous transmission is, however, a fast and reliable method of data transfer when you absolutely need to know that a message was received by the recipient, or when the sending task must have a response before it can continue processing.

3.2. Asynchronous data transmission

An asynchronous data transmission involves a mechanism called a queue. In general, a queue is a service which temporarily holds messages destined for a receiving task until that task is ready to process them. The sending task passes messages off to the queue and does not wait for a response from the receiver. The queue guarantees that if it accepts a message then that message will be delivered. Cogent's asynchronous messaging is implemented through a queue administrator, called Cascade QueueServer, or **qserve**.

Figure 3-2. Asynchronous data transmission

In the example above, when task 1 sends a message to task 2 it does so through the queue administrator. Task 1 transmits the message to the queue administrator, which replies immediately with an acknowledgment that it has held the message for delivery pending a request from task 2. Task 1 can now continue with its other duties without having to wait for task 2 to process the message.

The queue administrator then sends a *signal* to task 2 telling it that there is a message waiting for it in the queue. A signal is a special operating system mechanism that cannot block on another task and in this case acts like a flag to indicate to task 2 that there is a message waiting. This means that the queue program is never blocked waiting for a reply from a task that it has informed about a message waiting.

Task 2 sends a request for the waiting message and receives the data in a reply from the queue administrator.

It should be noted that the queue administrator never initiates a message transaction, and due to the nature of the send/receive/reply mechanism it will never block while transmitting a reply. Thus a queue administrator in this scenario will always be available to receive messages and the originator of an asynchronous message will never block.

There are a number of disadvantages to using asynchronous message passing. The sender and receiver in an asynchronous transaction must agree upon a queue name in order for the queue administrator to correctly route a message. This requires either hard-coded queue names, command-line arguments, or a queue-aware name server so tasks can discover the destination queue names. The Cascade DataHub uses a queue-aware name server called **nserve** to provide maximum flexibility in asynchronous connections.

A second concern is that because asynchronous transmission introduces additional message passing overhead into the transmission mechanism, the speed of asynchronous transmission will tend to be slower than that of synchronous data transfer. Perhaps the biggest drawback of the asynchronous transmission method is that there must be a method of dealing with overflows in the queue. If the receiving task fails to collect its messages from the queue administrator, or if it cannot keep up with the rate at which messages are being sent to the queue administrator, then eventually messages will build up and the queue administrator will run out of buffer space. At this point, the queue administrator must refuse further incoming messages until the receiver has cleared some of its pending messages. The sending task must include a contingency plan for undeliverable messages.

A sending task may react in several ways to a full queue situation:

1. **Throw away data that cannot be put on the queue.** This is both the most common and least acceptable response to a full queue. The most recent data will be lost, and old data currently on the queue will be retained. Once the condition causing the full queue has been cleared, the most recent data will be unavailable. In the case of process control systems, this scenario will fail to transmit the most recent process changes. Obviously, this method is not attractive when you consider the potential loss of state change information and critical alarm data.
2. **Throw away the oldest data on the queue.** This method is only viable if the queue is held internally to the sending task. In most cases, a task cannot walk the queue as it is provided by an external program or operating system facility. Even if the queue is available to the sender, there is no guarantee that the oldest data is the least important. In general, eliminating data solely on the basis of age is unacceptable. (In the case of the QNX queue administrator, **Mqueue**, this method of dealing with a full queue is not an option, as the queue is held externally to the sender.)
3. **Throw away the oldest duplicate data on the queue.** As above, this option is only available if the queue is internal to the sender. In addition, it implies that the data in the queue is of a known type and format, and can be examined for its similarity to new data. In the case of process control points, this is generally true, though it means that every point being transmitted must be compared to every queued message to determine whether it should replace a current message or go to the end of the queue. Depending on the implementation, this mechanism may not preserve time ordering in the queued data. The Cascade DataHub uses a generalized external queue mechanism, and so does not provide this type of data reduction.

4. **Try to send the data again later.** If a task cannot deliver a message to the queue immediately, then it has the option of holding the message for later transmission. Once space is available in the queue, the message can be re-transmitted. This method actually means that the sender is implementing its own internal queue, duplicating the work of the queue facility, and so is subject to all of the same concerns regarding full buffers and data reduction. In general, this approach by itself is entirely useless, and is exactly equivalent to enlarging the size of the queue administrator's buffers. When used in combination with one of the other methods mentioned here, this approach could be very effective.

5. **Try to send the data again later, throwing away old duplicate data from the retry queue only.** This is the method used by the Cascade DataHub to deal with undeliverable data. So long as the receiver is capable of keeping up with the data transmission rate, all point changes are deliverable. If the receiver allows its queue to fill, then the Cascade DataHub will flag the point as pending delivery to the recipient and will attempt to deliver the most recent value for that point as soon as there is room in the queue. If another change to a pending point is received by the DataHub, then the previous value is overwritten, so that when room in the queue becomes available, the new value is transmitted. In addition, the Cascade DataHub packages many point changes into a single message whenever there are several pending points to a task, thereby increasing the bandwidth of the queue administrator during periods of high load. In this way, the Cascade DataHub will transmit all point changes to any receiver that can handle the data rate, and guarantees that even slow receivers (such as GUI applications and applications on dial-up lines) will always receive the most recent point values even if they miss some intermediate values.

3.3. Cascade DataHub data transmission

The Cascade DataHub works in conjunction with the **qserve** (queue manager) program to provide asynchronous data transmission between tasks.

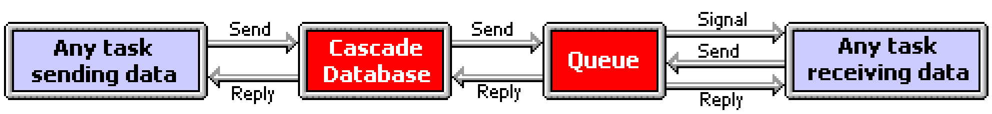


Figure 3-3. Cascade DataHub Data Transmission

Any number of tasks can write data directly to the Cascade DataHub, because it is a non-blocking program that is always ready to receive a message.

When a program wishes to read the point value from the DataHub, the task registers for exceptions for that point. Then, whenever the DataHub receives a new value for a particular data point, it immediately sends a message to the queue for each client program that has registered an interest in that data point. The queue manager then sends a signal to the client program, informing the client that there is a message waiting on its queue. The client program then sends a message to the queue manager requesting the message. The queue manager sends the new value for the data point in the reply to the client.

There are three ways in which a client can register for exceptions from the DataHub.

1. **A client registers for exceptions on a point-by-point basis.** Each time one of the registered points changes value in the DataHub, a message is sent to the client.

2. **A client can register for exceptions for all points currently in the DataHub.** Any change to the value of a data point that is placed in the DataHub after this registration will not be sent to the client.

3. **A client can register for exceptions for any point that ever appears in the DataHub.** All data change events in the DataHub will be broadcast to the client even if the point is created in the DataHub subsequent to the client's registration. When a DataHub client tries to register for exceptions for points that do not exist in the DataHub, the point is created in the DataHub and any further value changes are reported. The number of points in the DataHub is only limited by system resources. The DataHub has been tested with over 65,000 points and simply grows in size to meet the need of the application.

Appendix A. GNU General Public License

GNU General Public License

Version 2, June 1991

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change free software - to make sure the software is free for all its users. This General Public License applies to most of the Free Software Foundation's software and to any other program whose authors commit to using it. (Some other Free Software Foundation software is covered by the GNU Library General Public License instead.) You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs; and that you know you can do these things.

To protect your rights, we need to make restrictions that forbid anyone to deny you these rights or to ask you to surrender the rights. These restrictions translate to certain responsibilities for you if you distribute copies of the software, or if you modify it.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must give the recipients all the rights that you have. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

We protect your rights with two steps:

1. copyright the software, and
2. offer you this license which gives you legal permission to copy, distribute and/or modify the software.

Also, for each author's protection and ours, we want to make certain that everyone understands that there is no warranty for this free software. If the software is modified by someone else and passed on, we want its recipients to know that what they have is not the original, so that any problems introduced by others will not reflect on the original authors' reputations.

Finally, any free program is threatened constantly by software patents. We wish to avoid the danger that redistributors of a free program will individually obtain patent licenses, in effect making the program proprietary. To prevent this, we have made it clear that any patent must be licensed for everyone's free use or not licensed at all.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

Section 0

This License applies to any program or other work which contains a notice placed by the copyright holder saying it may be distributed under the terms of this General Public License. The "Program", below, refers to any such program or work, and a “work based on the Program” means either the Program or any derivative work under copyright law: that is to say, a work containing the Program or a portion of it, either verbatim or with modifications and/or translated into another language. (Hereinafter, translation is included without limitation in the term “modification”.) Each licensee is addressed as “you”.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running the Program is not restricted, and the output from the Program is covered only if its contents constitute a work based on the Program (independent of having been made by running the Program). Whether that is true depends on what the Program does.

Section 1

You may copy and distribute verbatim copies of the Program’s source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and give any other recipients of the Program a copy of this License along with the Program.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Section 2

You may modify your copy or copies of the Program or any portion of it, thus forming a work based on the Program, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. You must cause the modified files to carry prominent notices stating that you changed the files and the date of any change.

b. You must cause any work that you distribute or publish, that in whole or in part contains or is derived from the Program or any part thereof, to be licensed as a whole at no charge to all third parties under the terms of this License.

c. If the modified program normally reads commands interactively when run, you must cause it, when started running for such interactive use in the most ordinary way, to print or display an announcement including an appropriate copyright notice and a notice that there is no warranty (or else, saying that you provide a warranty) and that users may redistribute the program under these conditions, and telling the user how to view a copy of this License. (Exception: If the Program itself is interactive but does not normally print such an announcement, your work based on the Program is not required to print an announcement.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Program, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Program, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Program.

In addition, mere aggregation of another work not based on the Program with the Program (or with a work based on the Program) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

Section 3

You may copy and distribute the Program (or a work based on it, under Section 2 in object code or executable form under the terms of Sections 1 and 2 above provided that you also do one of the following:

a. Accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

b. Accompany it with a written offer, valid for at least three years, to give any third party, for a charge no more than your cost of physically performing source distribution, a complete machine-readable copy of the corresponding source code, to be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange; or,

c. Accompany it with the information you received as to the offer to distribute corresponding source code. (This alternative is allowed only for noncommercial distribution and only if you received the program in object code or executable form with such an offer, in accord with Subsection b above.)

The source code for a work means the preferred form of the work for making modifications to it. For an executable work, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the executable. However, as a special exception, the source code distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

If distribution of executable or object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place counts as distribution of the source code, even though third parties are not compelled to copy the source along with the object code.

Section 4

You may not copy, modify, sublicense, or distribute the Program except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense or distribute the Program is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

Section 5

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Program or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Program (or any work based on the Program), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Program or works based on it.

Section 6

Each time you redistribute the Program (or any work based on the Program), the recipient automatically receives a license from the original licensor to copy, distribute or modify the Program subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties to this License.

Section 7

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Program at all. For example, if a patent license would not permit royalty-free redistribution of the Program by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Program.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system, which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

Section 8

If the distribution and/or use of the Program is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Program under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

Section 9

The Free Software Foundation may publish revised and/or new versions of the General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of this License, you may choose any version ever published by the Free Software Foundation.

Section 10

If you wish to incorporate parts of the Program into other free programs whose distribution conditions are different, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY Section 11

BECAUSE THE PROGRAM IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE

PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

Section 12

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program’s name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This program is free software; you can redistribute it and/or modify it under the terms of the GNU General Public License as published by the Free Software Foundation; either version 2 of the License, or (at your option) any later version.

This program is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU General Public License for more details.

You should have received a copy of the GNU General Public License along with this program; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

If the program is interactive, make it output a short notice like this when it starts in an interactive mode:

Gnomovision version 69, Copyright (C) year name of author Gnomovision comes with ABSOLUTELY NO WARRANTY; for details type “show w”. This is free software, and you are welcome to redistribute it under certain conditions; type “show c” for details.

The hypothetical commands “show w” and “show c” should show the appropriate parts of the General Public License. Of course, the commands you use may be called something other than “show w” and “show c”; they could even be mouse-clicks or menu items--whatever suits your program.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the program, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the program “Gnomovision” (which makes passes at compilers) written by James Hacker.

<signature of Ty Coon>, 1 April 1989 Ty Coon, President of Vice

This General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Library General Public License instead of this License.

Appendix B. GNU Lesser General Public License

GNU Lesser General Public License

Version 2.1, February 1999

Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method:

1. we copyright the library, and
2. we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the *Lesser* General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

Section 0

This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

Section 1

You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

Section 2

You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a. The modified work must itself be a software library.

b. You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c. You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d. If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

   (For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

Section 3

You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

Section 4

You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

Section 5

A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

Section 6

As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a. Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable “work that uses the Library”, as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)
b. Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user’s computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.
c. Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.
d. If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.
e. Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the “work that uses the Library” must include any data and utility programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

Section 7

You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a. Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.
b. Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

Section 8

You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

Section 9

You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

Section 10

Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

Section 11

If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

Section 12

If the distribution and/or use of the Library is restricted in certain countries either by patents or by copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

Section 13

The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and “any later version”, you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

Section 14

If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

NO WARRANTY Section 15

BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

Section 16

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Libraries

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the library’s name and a brief idea of what it does.> Copyright (C) <year> <name of author>

This library is free software; you can redistribute it and/or modify it under the terms of the GNU Lesser General Public License as published by the Free Software Foundation; either version 2.1 of the License, or (at your option) any later version.

This library is distributed in the hope that it will be useful, but WITHOUT ANY WARRANTY; without even the implied warranty of MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the GNU Lesser General Public License for more details.

You should have received a copy of the GNU Lesser General Public License along with this library; if not, write to the Free Software Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA 02110-1301 USA

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a “copyright disclaimer” for the library, if necessary. Here is a sample; alter the names:

Yoyodyne, Inc., hereby disclaims all copyright interest in the library ’Frob’ (a library for tweaking knobs) written by James Random Hacker.

<signature of Ty Coon>, 1 April 1990 Ty Coon, President of Vice

That’s all there is to it!

I. Utilities

Table of Contents

The Cascade DataHub ships with a number of utilities, which are installed on the system. This reference section describes how to use these utilities.

datahub

datahub — starts the Cascade DataHub.

Synopsis

datahub [-aDhstUvVX][-b *size*][-d *domain*][-f *file*][-H *home_path*][-l *file*] [-m *port*][-M *address*][-n *domain*][-p *port*][-q *queue*]

Arguments

-a

Transmit all point messages to all registered clients, even if the value does not change.

-b *size*

The maximum message buffer size.

-d *domain*

The domain name for this DataHub. This option can be used multiple times to get multiple domains on a single DataHub.

-D

Do not detach from the controlling tty. Normally the DataHub will detach itself and become immune to interrupts and termination on the controlling tty. If this option is used, then an *&* is necessary to run **datahub** in the background.

-f *file*

Load this configuration file.

-h

Print a help message showing a summary of all these arguments.

-H *home_path*

The full path to the directory that will contain the configuration and license files. This takes precedence over `-U`. If the directory cannot be found or created, the files will be stored in the installation directory.

-l *file*

Log messages to this file.

-m *port*

Acting as a TCP slave, attach to a TCP master on this port or service. The port is the matching port number of the master, usually `4600`;

-M *address*

Acting as a TCP slave, attach to a TCP master on this host. The address is a machine name, such as `developers.cogentrts.com` or a machine address, such as `192.168.3.15`.

-n *domain*

Acting as a TCP slave, mirror this domain from the TCP master. The named domain on the master will be mirrored to a domain of the same name in the slave.

-p *port*

Act as a TCP master and listen on this port/service.

`-q` *`queue`*

Specify an alternate queue name for this DataHub. Normally **datahub** chooses its own queue name to be unique on the network.

`-s`

Synchronized: The DataHub will ignore changes to a point if the point's current timestamp is more recent.

`-t`

Automatically generate a timestamp on unstamped points.

`-U`

The DataHub should NOT create a directory within the user's personal `Application Data` directory to store the configuration and license files, but rather in the application installation directory. This has a lower precedence than `-H`.

`-v`

Generate copious debugging information to the standard output. (Implies use of `-D`).

`-V`

Print the version number.

`-X`

Exit immediately (usually used with `-V`).

Returns

On success, nothing; on error, a message.

Description

The **datahub** command starts the DataHub. The DataHub does not use a configuration file since it is configured by its clients while running. Except where noted, the command line options apply to Cascade DataHub in Windows, Linux, or QNX, as well as Cascade Connect.

Dependencies

qserve, **nserve**

See Also

Using the Cascade DataHub, **dhview**, **phdhview**, **xdhview**

Example

```
[sh]$  datahub
```

Starts the DataHub in the domain `default`.

```
[sh]$  datahub -d control
```

Starts the DataHub in the domain `control`.

dhview

`dhview` — a console-based viewer for the Cascade DataHub.

Synopsis

dhview `[-hvVX]` `[-d` *`domain`*`]`

Arguments

`-d` *`domain`*

Specify the domain of the DataHub to view, defaults to '`default`'.

`-h`

Print a help message and exit.

`-v`

Generate debugging output.

`-V`

Print the version number.

`-X`

Exit immediately (usually used with `-V`).

Description

The **dhview** utility presents a text-based window containing a list view of the points within the Cascade DataHub for a specified domain. The **dhview** display is updated as points within the DataHub change.

Returns

On success, nothing; on error, a message.

Dependencies

qserve, **nserve**, **datahub**

See Also

Console Mode in Using the Cascade DataHub Viewer, **phdhview**, **xdhview**

Example

```
[sh]$  dhview
```

Displays the DataHub from the domain `default`.

```
[sh]$  dhview -d control
```

Displays the DataHub from the domain `control`.

phdhview

`phdhview` — a Photon-based viewer for the Cascade DataHub.

Synopsis

phdhview [`-hvVX`] [`-d` *`domain`*]

Arguments

`-d` *`domain`*

Specify the domain of the DataHub to view, defaults to '`default`'.

`-h`

Print a help message and exit.

`-v`

Generate debugging output.

`-V`

Print the version number.

`-X`

Exit immediately (usually used with `-V`).

Returns

On success, nothing; on error, a message.

Description

The **phdhview** utility presents a Photon window containing a list view of the points within the Cascade DataHub for a specified domain. The **phdhview** display is updated as points within the DataHub change.

Dependencies

qserve, **nserve**, **datahub**

See Also

QNX Photon Mode in Using the Cascade DataHub Viewer, **dhview**, **xdhview**

Example

```
[sh]$  phdhview
```

Views the DataHub for the domain `default`.

```
[sh]$
    phdhview -d control
```

Views the DataHub for the domain `control`.

readpt

`readpt` — reads a point from the Cascade DataHub.

Synopsis

readpt `[-bh]` `[-d` *`domain`*`]` *`pointname`*...

Arguments

`pointname`

The name of the point to retrieve information for. Multiple points can be specified.

`-b`

Brief output. Print only the point name and value.

`-d` *`domain`*

The domain name of the desired datahub. This replaces the default, which is "`default`".

`-h`

Print a help message and exit.

Returns

On success, information on the requested points; on error, a message.

Description

The **readpt** utility will retrieve the value of a point from any available Cascade DataHub. If a point cannot be found a new point will be created in the DataHub it was queried from. The domain can be specified with `-d`, or by qualifying the `pointname`, as in *`domain`*`:`*`pointname`* (the latter technique overriding the former).

The source code for this utility is normally installed in the `/usr/cogent/src/datahub` directory.

Dependencies

nserve, **datahub**

See Also

datahub, **writept**

Example

```
[sh]$  readpt LIC02_sp
  Point: LIC02_sp
  Value: 55
  Conf: 100
  Lock: 0
  Secur: 0

[sh]$  readpt -d control m23onoff
  Point: m23onoff
  Value: 1
  Conf: 100
```

```
  Lock: 0
  Secur: 0

[sh]$  readpt -b control:m23onoff
  Point: m23onoff
  Value: 1
```

waiter

waiter — registers for exceptions with a Cascade DataHub and displays values as updates are received.

Synopsis

waiter [-hv] [-d *domain*] [-q *queuename*] [*pointname*...]

Arguments

-d *domain*

The domain name of the DataHub to write the point to.

-h

Print a help message and exit.

-q *queuename*

The name of the queue for point changes.

-v

Print debugging information.

pointname

The name of point(s) to register for exceptions. This argument is optional. If not supplied, all points in the DataHub will be displayed.

Returns

On success, prints a message indicating that the points named on the command line have been registered and the values of registered points when they change. On error, an error message.

Description

The **waiter** utility will register for exceptions on the named DataHub, or on the default DataHub if -d is not specified. If points are named on the command line, then only those points are expected. If no points are specified, then all points in the DataHub are registered, and any points created on the DataHub in future are also registered. Whenever a value change occurs on any registered point, **waiter** will generate a message to the console with information for that point.

If a point name is passed using the *domain*:*pointname* syntax then alternate domains can be registered without having to use the -d option. Points from multiple domains can be watched simultaneously using this notation. For example, this:

```
[sh]$ waiter mixer:Mixer_1_Weight
```

is the same as this:

```
[sh]$ waiter -d mixer Mixer_1_Weight
```

The source code for this utility is normally installed in the /usr/cogent/src/datahub directory.

Dependencies

qserve, **nserve**, **datahub**

See Also

Using the Cascade DataHub, **datahub**, **readpt**, **writept**

Example

```
[sh]$  waiter LIC02_sp LIC02_pv
```

Registers for exceptions on the points `LIC02_sp` and `LIC02_pv` in the `default` DataHub.

```
[sh]$  waiter -d control
```

Registers for exceptions for all points in the DataHub `control`.

If you have only two points, `p3` and `p4` in the database with values of 15 and 19 respectively, and you start **waiter** with no options, you will get output like this:

```
[sh]$  waiter
Point: p3
Value: 15
Conf:  100, Lock: no, Time: Fri Sep 14 2001 14:56:43.888, Security: 0
Point: p4
Value: 19
Conf:  100, Lock: no, Time: Fri Sep 14 2001 14:56:25.529, Security: 0
```

Should the value of `p4` change to 25, you would see the following on your console:

```
Point: p4
Value: 25
Conf:  100, Lock: no, Time: Fri Sep 14 2001 14:57:23.018, Security: 0
```

writept

`writept` — writes a point to the Cascade DataHub.

Synopsis

writept `[-frils][-d` *`domain`*`][-S` *`security`*`]`*`pointname pointvalue`*

Arguments

`pointname`

The name of the Cascade DataHub point to write to.

`pointvalue`

The value to assign to this point.

`-d` *`domain`*

Domain name for the DataHub to write the point to.

`-f`

Write the point as a floating point number.

`-i`

Write the point as a short integer.

`-l`

Write the point as a long integer.

`-r`

Same as `-f`.

`-s`

Write the point as a character string.

`-S` *`security`*

Set the security level for writing.

Returns

On success, nothing; on error, a message.

Description

The **writept** utility will write or create a value in any accessible Cascade DataHub. Alternate domains can be written to without having to use the **`-d`** option by passing point names with the **`domain:pointname`** syntax.

Strings containing spaces and special characters must be escaped from the shell appropriately. The point type will be guessed if not specified. The order of guessing is: long, float, string (the default). The guess is considered correct if the entire argument is converted. Long can be specified in standard C style, for example `0x5f` for hex, `0647` for octal, etc.

The source code for this utility is normally installed in the `/usr/cogent/src/datahub` directory.

Dependencies

qserve, **nserve**, **datahub**

See Also

datahub, **readpt**

Example

```
[sh]$  writept LIC02_sp 55
```

Writes to the point `LIC02_sp` the value of `55` in the DataHub `default`.

```
[sh]$  writept -d control m38onoff 1
```

Writes to the point `m38onoff` the value `1` in the DataHub `control`.

xdhview

`xdhview` — an X Windows-based viewer for the Cascade DataHub.

Synopsis

xdhview [`-hvVX`] [`-d` *`domain`*]

Arguments

`-d` *`domain`*

Specify the domain of the DataHub to view, defaults to ’`default`’.

`-h`

Print a help message and exit.

`-v`

Generate debugging output.

`-V`

Print the version number.

`-X`

Exit immediately (usually used with `-V`).

Returns

On success, nothing; on error, a message.

Description

The **xdhview** utility uses X Windows to present a window containing a list view of the points within the Cascade DataHub for a specified domain. The **xdhview** display is updated as points within the DataHub change.

Dependencies

qserve, **nserve**, **datahub**

See Also

X Windows Mode in Using the Cascade DataHub Viewer, **dhview**, **phdhview**

Example

```
[sh]$ xdhview
```

Views the DataHub from the domain `default`.

```
[sh]$ xdhview -d control
```

Views the DataHub from the domain `control`.

Index

Colophon

This book was produced by Cogent Real-Time Systems, Inc. from a single-source group of SGML files. Gnu Emacs was used to edit the SGML files. The DocBook DTD and related DSSSL stylesheets were used to transform the SGML source into HTML, PDF, and QNX Helpviewer output formats. This processing was accomplished with the help of OpenJade, JadeTeX, Tex, and various scripts and makefiles. Details of the process are described in our book: Preparing Cogent Documentation, which is published on-line at
`http://developers.cogentrts.com/cogent/prepdoc/book1.html`.

Text written by Andrew Thomas, Paul Benford, and Bob McIlvride.
Illustrations by Paul Benford.