

MVD message test system. 29/09/02 (CY).

Introduction

A system for testing message passing through the Hub is described.

Contents:

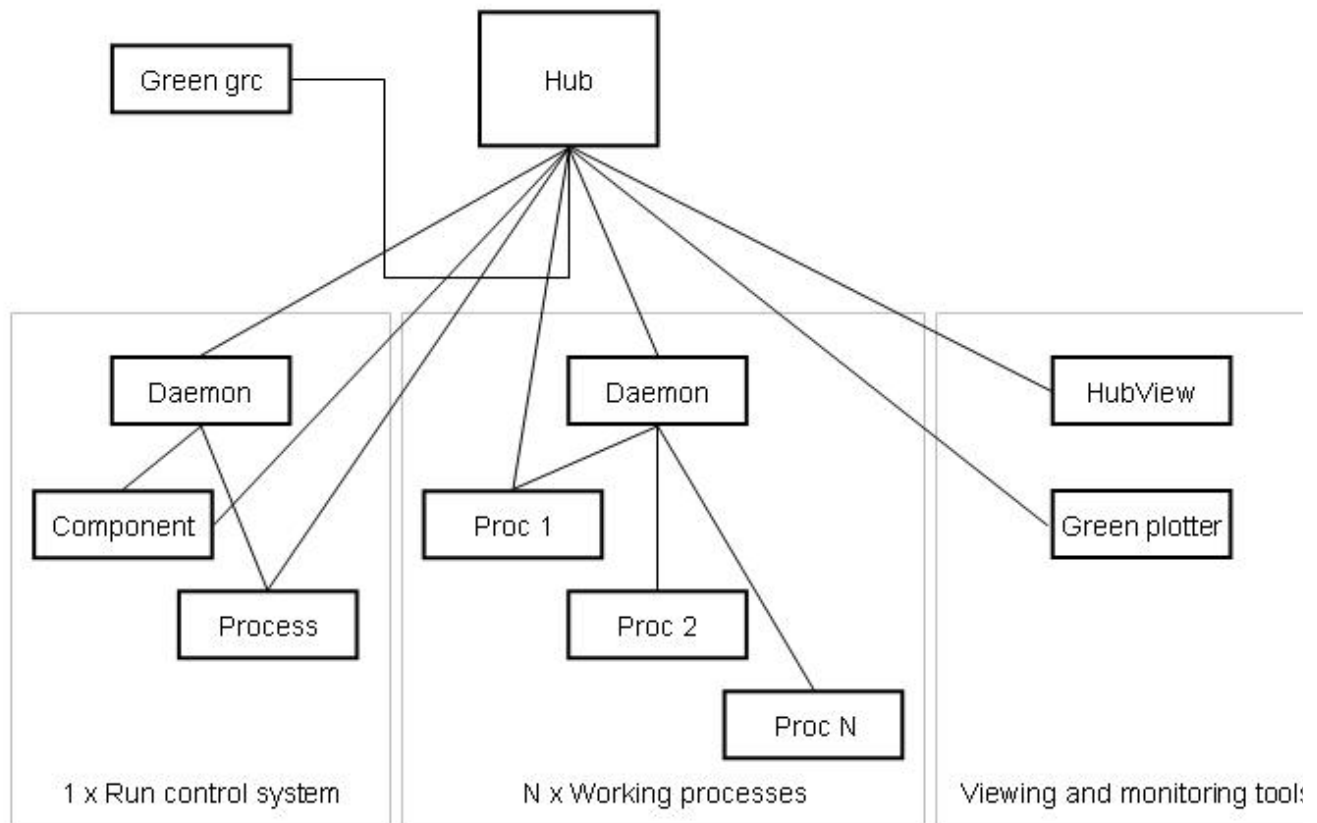
- [Purpose](#)
- [Contents of test/](#)
- [Test configuration](#)
- [Run configuration](#)
- [Example test run](#)
 - [Make](#)
 - [Starting daemons](#)
 - [Start the Hub](#)
 - [Start the run control interface](#)
 - [Running processes](#)
 - [Check the test](#)
- [Useful test clients and scripts](#)
 - [Busy writer](#)
 - [Lazy reader](#)
 - [Reuse connection](#)

Purpose

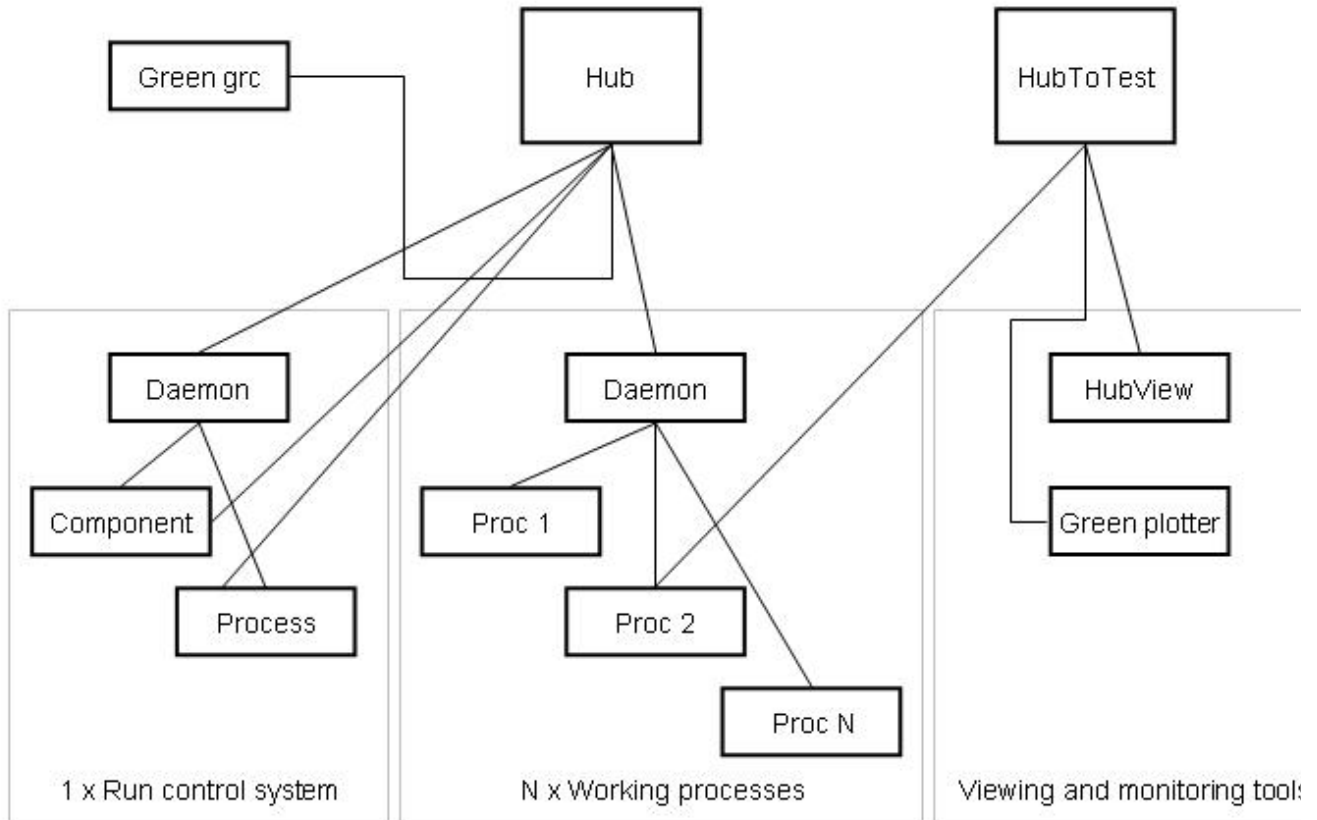
The purpose of the test system is to provide a framework for testing:

1. modifications to the message definitions,
2. new implementations of Hub client processes, and
3. new implementations of the Hub

under realisting conditions (concurrent message passing, load variation, run control, etc.).
The configurations used for 1-2 and 3 are shown schematically in below.



Message and new client test configuration.



Hub test configuration.

The principle components are:

- **Hub** - A working Hub.
- **HubToTest** - The Hub to be tested.
- **Run control system** - A run control system connected to Hub.
- **Green grc** - A run controller (grc or rc_local_curses.exe) connected to Hub.
- **Working processes** - One or more sets of working processes which connect to the Hub being tested, but controlled by a daemon connected to Hub.
- **Viewing and monitoring tools** - Processes connected to the Hub or HubToTest which monitor the performance of the test.

[Contents of test/](#)

The source of the MVD message passing system includes a `test/` directory holding a self-contained test system which can be run concurrently on the same host as the MVD system without affecting it.

The test directory contains the following files and directories:

- **test/**
 - **Makefile** - make file to remove or make the test system.
 - ***.monitor_org** - prototype monitor files defining process parameters required by the daemon.
 - ***.daemon_org** - prototype daemon files listing the monitor files required by a daemon.
 - ***.script_org** - prototype scripts used to start daemons controlling the RC system or working processes.
 - ***.c** - Hub test clients.
- **test/config/**
 - **monitor_awkscr** - awk script defining Hub addresses.
 - **monitor_awkscr_A** - awk script defining the Hub addresses (port@host) needed for the simplified test (OK_HUB_ADDR == TEST_HUB_ADDR).
 - **monitor_awkscr_B** - awk script defining the Hub addresses (port@host) needed for Hub tests (OK_HUB_ADDR != TEST_HUB_ADDR).
 - ***.monitor** - gawk generated from test/*.monitor_org by make.
 - ***.daemon** - gawk generated from test/*.daemon_org by make.
- **test/std/**
 - ***.std** - state transition diagrams used by run control and referenced from runtime files.
- **test/runtype/**
 - ***.runtype** - run configuration files used by run control
- **test/work/**
 - ***** - default directory of rc_component.exe and prc_control.exe.

Test configure

To configure the test system perform the following operations:

1. cd test/
2. cp monitor_awkscr_Xmonitor_awkscr
3. make clean
4. make

where X is either A or B, see above. Amongst other things make runs gawk on the *.*_org files in test/ replacing OK_HUB_ADDR and TEST_HUB_ADDR strings with the values specified in the awk scripts.

Run configure

The run configuration is defined by the contents of the runtime file selected by the run controller. Note that the run configuration is independent of the test configuration.

Two run configurations are currently provided:

- **hub_busy_quick_lazy.runtype**

which tests Hub throughput rates, bandwidths and performance w.r.t. well behaved (quick) and slow (lazy) receiving clients.

- **hub_plot_fill_send.runtype**
which tests throughput of histogram plots.

The runtype/* .runtype and std/* .std files provided have the same format as those documented in the MVD run control description. New run configurations must be added into the appropriate directories.

The runtype file provides a mechanism of changing test parameters associated with the working processes dynamically by prompting for the values via the run controller. Unfortunately grc must be restarted after replying as there seems to be a problem associated with setting parameters in this way - the reply values are actually set ?!

Example test run

This section contains a blow-by-blow account of how to make and run the monitor_awkscr_A test configuration with the hub_plot_fill_send.runtype configuration.

Make

```
youngman@cypc2:~> cd ~/toinfo/test
youngman@cypc2:~/toinfo/test> cp config/monitor_awkscr_A config/monitor_awkscr
youngman@cypc2:~/toinfo/test> make clean
rm -f temp core *~ /home/youngman/mvddaq/src/bin/busy_writer.exe /home/youngman/mvddaq/src/bin/lazy_re
home/youngman/mvddaq/src/obj/busy_writer.o /home/youngman/mvddaq/src/obj/lazy_reader.o /home/youngmar
/bin/reuse_connection.script /home/youngman/mvddaq/src/bin/start_hub_test_ctr.script /home/youngman/mv
n/start_hub_test_environment.script ./config/busy_writer.monitor ./config/lazy_reader.monitor ./config
r.monitor ./config/plot_sender.monitor ./config/quick_reader.monitor ./config/test_prc_control.monitor
est_rc_component.monitor ./config/hub_test_ctr.daemon ./config/hub_test_environment.daemon *~
youngman@cypc2:~/toinfo/test> make
gcc -c busy_writer.c -DSUSE -DUNIX -D_REENTRANT -I/home/youngman/mvddaq/src/include -o /home/youngma
c/obj/busy_writer.o
gcc /home/youngman/mvddaq/src/obj/busy_writer.o -L/home/youngman/mvddaq/src/lib -lcy -o /home/youngn
rc/bin/busy_writer.exe
gcc -c lazy_reader.c -DSUSE -DUNIX -D_REENTRANT -I/home/youngman/mvddaq/src/include -o /home/youngma
c/obj/lazy_reader.o
gcc /home/youngman/mvddaq/src/obj/lazy_reader.o -L/home/youngman/mvddaq/src/lib -lcy -o /home/youngn
rc/bin/lazy_reader.exe
gawk -f ./config/monitor_awkscr reuse_connection.script_org > /home/youngman/mvddaq/src/bin/reuse_conr
pt
chmod +x /home/youngman/mvddaq/src/bin/reuse_connection.script
gawk -f ./config/monitor_awkscr start_hub_test_ctr.script_org > /home/youngman/mvddaq/src/bin/start_hu
script
chmod +x /home/youngman/mvddaq/src/bin/start_hub_test_ctr.script
gawk -f ./config/monitor_awkscr start_hub_test_environment.script_org > /home/youngman/mvddaq/src/bin/
est_environment.script
chmod +x /home/youngman/mvddaq/src/bin/start_hub_test_environment.script
gawk -f ./config/monitor_awkscr busy_writer.monitor_org > config/busy_writer.monitor
gawk -f ./config/monitor_awkscr lazy_reader.monitor_org > config/lazy_reader.monitor
gawk -f ./config/monitor_awkscr plot_filler.monitor_org > config/plot_filler.monitor
gawk -f ./config/monitor_awkscr plot_sender.monitor_org > config/plot_sender.monitor
gawk -f ./config/monitor_awkscr quick_reader.monitor_org > config/quick_reader.monitor
gawk -f ./config/monitor_awkscr test_prc_control.monitor_org > config/test_prc_control.monitor
gawk -f ./config/monitor_awkscr test_rc_component.monitor_org > config/test_rc_component.monitor
```

```
gawk -f ./config/monitor_awkscr hub_test_ctr.daemon_org > config/hub_test_ctr.daemon
gawk -f ./config/monitor_awkscr hub_test_environment.daemon_org > config/hub_test_environment.daemon
rm /home/youngman/mvddaq/src/obj/busy_writer.o /home/youngman/mvddaq/src/obj/lazy_reader.o
youngman@cypc2:~/toinfo/test>
```

Starting daemons

First the run control daemon.

```
youngman@cypc2:~> start_hub_test_ctr.script
starting test ctr host cypc2 service dae_test_ctr file hub_test_ctr.daemon
```

The, say, two working process daemons. Note that the HOST environmental has to be set to a character string ending the end number of which is used to distinguish between the work systems.

```
youngman@cypc2:~> export HOST=abc01
youngman@cypc2:~> start_hub_test_environment.script
starting test environment host abc01 service dae_test_env_abc01 file hub_test_environment.daemon
youngman@cypc2:~> export HOST=abc02
youngman@cypc2:~> start_hub_test_environment.script
starting test environment host abc02 service dae_test_env_abc02 file hub_test_environment.daemon
youngman@cypc2:~>
```

```
youngman@cypc2:~> ps afx
1392 pts/3    S      0:00 /home/youngman/mvddaq/src/bin/daemon.exe dae_test_ctr 13333 hub_test_ctr.dae
 1393 pts/3    S      0:00 \_ /home/youngman/mvddaq/src/bin/rc_component.exe 13333 component process
 1394 pts/3    S      0:00 \_ /home/youngman/mvddaq/src/bin/prc_control.exe process component dae* 1:
 1396 pts/3    S      0:00 /home/youngman/mvddaq/src/bin/daemon.exe dae_test_env_abc01 13333 hub_test_
 1398 pts/3    S      0:00 /home/youngman/mvddaq/src/bin/daemon.exe dae_test_env_abc02 13333 hub_test_
```

Start the Hub

```
youngman@cypc2:~> which Hub.exe
/home/youngman/mvddaq/src/bin/Hub.exe
youngman@cypc2:~> Hub.exe 13333 >/dev/null
```

Start the run control interface

Make sure that the, possibly aliased, standard target is not used.

```
youngman@cypc2:~> alias grc
alias grc='echo using alias for green; cd ~/green; grc 19560'
youngman@cypc2:~> unalias grc
youngman@cypc2:~> cd ~/green
youngman@cypc2:~/green> grc 13333 >/dev/null
```

Grc when started requires that a runtime configuration be selected. Navigate to the test/runtype/*.runtype file required using the file selection box "ProcDefn".

Once the runtime configuration is sent the test run can be stopped and started by navigating through the state-transition diagram.

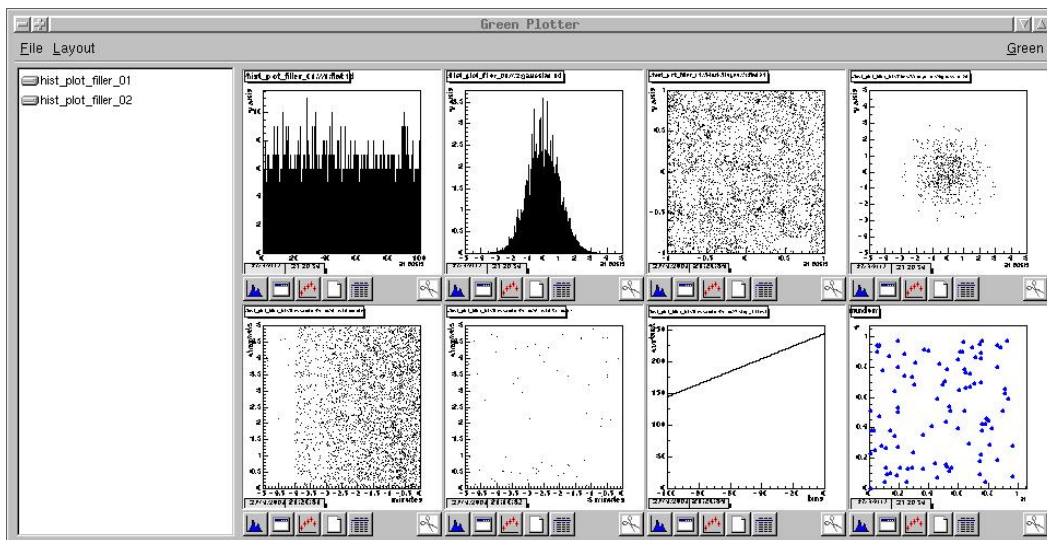

```

1393 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/rc_component.exe 13333 component process
1394 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/prc_control.exe process component dae* 13
1396 pts/3    S      0:00  /home/youngman/mvdda/src/bin/daemon.exe dae_test_env_abc01 13333 hub_test_
1492 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/plot_loop.exe PF01
1494 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/plot_server.exe hist_plot_filler_01 1333:
1500 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/simple_client.exe 13333 * 0
1398 pts/3    S      0:00  /home/youngman/mvdda/src/bin/daemon.exe dae_test_env_abc02 13333 hub_test_
1493 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/plot_loop.exe PF02
1495 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/plot_server.exe hist_plot_filler_02 1333:
1501 pts/3    S      0:00  \_ /home/youngman/mvdda/src/bin/simple_client.exe 13333 * 0

```

Check the test

In this example the plot testing runtime configurations was chosen. Use the right-hand grc pull down menu to select Gplotter and select the corresponding layout and check that plots are being produced.



Using gplotter to check the test run.

Useful test clients and scripts

This section outlines, briefly, some test/ processes and scripts which are useful when performing tests. Other useful processes, eg. `simple_client.exe`, `peek_forward.exe`, are documented elsewhere.

Busy writer

Busy writer as the name suggests is a Hub client which write messages to the Hub in a loop.

Usage: `busy_writer.exe public_service_name hub_addr timer_msec_between_msgs msg_size_words`

Lazy reader

Lazy reader is a Hub client which badly reads messages from the Hub in a loop.

Usage: lazy_reader.exe hub_addr name_match tag_match nr_of_messages_N_to_read secs_to_wait_between_Ns

Reuse connection

The reuse_connection.script makes repeated connection requests using peek_forward.exe.

Usage: reuse_connection.script loop_count