

D0 Data Logger

States:

Pre-initialized

DataLogger connects to COOR-DL and wait for COOR-DL to give it configuration parameters using "INIT" command. (For now, since there is not COOR-DL DataLogger starts the server and wait for COOR to connect).

In this state only two operations are accepted: COOR "INIT" and COOR "STOP"

Initialized

DataLogger starts the server for Collector/Routers and is waiting for connections to be open. The client connections to Alarm and Monitor Servers are also open in this step.

The parameters which can be set in this step:

- port for the main server, addresses for Alarm and Monitor Servers
- streams default parameters (those which are used for all streams)
- debug level

No events are accepted at this state.

Configured

The information about streams is loaded. All streams spooled by this logger have to be listed with their unique parameters. The default stream parameters as well as debug level and other configuration parameters can be modified.

No events are accepted unless other run is already running.

Running

After receiving "Start Run" command the current stream configuration is "copied" to the configuration for the given run and the run is put in the "ACTIVE" state. The Stream objects are created for each stream listed in the configuration. Each Stream object is a single thread and it has list of File objects. After that all events from this run are accepted and written to a file.

The File object has four states:

- OPEN events can be written
- END file is full but events from the range can be written
- CLOSED file is really closed – no more events
- ERROR

There can be more then one run running at the same time.

D0 Data Logger

Connections:

Event Server

This is the main server to receive events in the form of Event_Message. No other message types are read (other types of messages are skipped). In the current D0me there is no way to register for "other" messages to monitor them and report problem. Should we have something like that?

The new version of D0me will have states and statistics for all connections. This means that DataLogger (as well as other D0me using processes) don't have to create their own monitoring tools for connections.

To COOR(-DL)

Now this connection is the server connection for the DataLogger. The problem is that in current D0me it is hard to implement the other way. But if the new D0me has identification and automatic reconnection build in we can think about doing this as a client connection (DataLogger->COOR-DL). There are number of pluses if this could be a client connection:

- DL has to run second server with the separate ip address just for this connection. One thread less and simpler code.
- All applications will use one port number instead of a number of port numbers as servers. This can be beneficial for testing, start new COOR-DL with different port number and all processes will connect to it and then follow COOR-DL commands.

To Alarm Server

This will be the client connection with automatic reconnect.

To Monitor

This will be the client connection with reconnect. The "information" from the DL could be sent periodically or per request or both.

D0 Data Logger

Recovery and Errors:

Server connections

There is no problem if the connection from C/R breaks in most of the cases. If the C/R doesn't die the reconnection will succeed and no events are lost. Even if the events are lost since they are lost randomly before the split or for some runs that we don't care it should still OK. In other case DL will expect command from COOR to "abort run" or something like this.

In new D0me we can think about sending information about process and then the reconnection can be really transparent. Do we need/what that?

Client connections COOR -DL(?) , Alarm, Monitor

This should be a problem at all and D0me should take care of reconnection.(?)

Some other questions:

- After a failure, should DataLogger quit or should it be put in the "INIT" or "CONFIG" state?
- I am planing to have two kinds of errors: ERROR – something bad but DataLogger can still run and FAILRE – panic. Which are which?
- Events which can't be written (files for this event range are already closed). At this moment I am planning to write this event to the "garbage collection" file for that stream but what should we do if the run isn't started or the stream doesn't exists.
- Disk full or other events writing problems. Currently DL will stop for any write failure. There should be a way to "pause" the system or just the DataLogger before the crush.
- Should we think about restarting DL after crush from the last state? This is not very hard to implement and can be much easier then asking COOR what to do after crush.
- What we can do to minimize loss of data in case of crush? Flush the files periodically and write metadata temporal files which can be used to recover partly written file.

Efficiency?