

NOvA Distributed Message Logging Requirements Document

Kurt Biery, Glenn Cooper, Steve Foulkes, Gerald M. Guglielmo,
Luciano Piccoli, Seangchan Ryu, Margaret E. Votava
Fermi National Accelerator Laboratory

NOVA-doc-xyz

(Last revision on 27-Mar-2006)

1	Introduction.....	2
1.1	Purpose.....	2
1.2	Scope.....	3
1.3	Rationale.....	3
1.4	Terminology.....	3
2	Overview.....	3
2.1	DAQ system.....	3
3	Requirements.....	3
3.1	Robustness and scalability.....	4
3.2	Client interface.....	4
3.2.1	API calls should be non-blocking.....	4
3.2.2	Supported languages.....	4
3.2.3	Aspect-oriented support?.....	Error! Bookmark not defined.
3.3	Servers/concentrators.....	4
3.4	Message destinations.....	5
3.4.1	File destinations.....	5
3.4.2	Secondary process destinations.....	5
3.4.3	Database destinations.....	5
3.5	Message filtering.....	5
3.5.1	At the display level.....	5
3.5.2	At the sending level.....	5
3.5.3	Smoothing of bursts.....	5
3.6	Message definition.....	5
3.7	Interface(s) to third party tools.....	6
3.8	Message-based actions.....	6
3.9	Web interface.....	Error! Bookmark not defined.
3.10	Historical analysis tools?.....	Error! Bookmark not defined.
3.11	Configuration.....	6
3.11.1	Remote Configuration.....	6
3.11.2	Persisting configurations.....	6
3.11.3	Dynamic reconfiguration.....	7
4	Testing – QA/QC.....	7

1 Introduction

1.1 Purpose

The purpose of this document is to define the message logging system for the NOvA Experiment and to present the requirements of its operation. The document is numbered in outline format, with *level-3 outline numbers corresponding to specific requirements or specifications.*

1.2 Scope

This document will present the requirements but will not specify how those requirements are to be met. In this sense, it is not a description of the intended implementation.

1.3 Rationale

Third-party systems may be available that meet some or all of these requirements. Candidate systems will be evaluated against these criteria. In the absence of a pre-packaged system that meets all of these requirements, a system that meets some of the requirements may be selected and custom extensions provided.

1.4 Terminology

Acronym	Description
DAQ	Data Acquisition System
Message	Information that reports on application state or exceptions

2 Overview

The message logging system provides the infrastructure for generating messages, transporting them to central locations for convenient access, displaying them for real time monitoring, and persisting them for later reference. Message contents follow a well-defined template, but within the template, allow for flexibility so that the addition of new messages to the system does not require the entire system to be rebuilt. The system is dynamic in the sense that bursts of messages can be smoothed out, destinations can be changed during running, and actions can be specified to occur in response to certain messages. Messages can be filtered based on user-specified criteria at the generation, transport, and display levels.

For some parts of the system, we want to be able to configure the message destination(s) for sources, but for displays, do we want to clients to be able to specify message filtering when they subscribe?

2.1 DAQ system

Describe the elements of the DAQ system here?

3 Requirements

The requirements in this section cover the following aspects of the message logging system:

- Generation
- Transport

- Reception
- Persistence
- Display
- Response (automatic actions based on specific messages)
- Contents
- Configuration
- Performance
- Filtering

3.1 Robustness and scalability

The system should be highly available and reliable. A small amount of message loss can occur in extremely disruptive failure modes, but the system should take reasonable steps to avoid message loss such as buffering messages to gracefully handle short network outages.

3.2 Scalability

For the full experiment, the system should support 1000-2000 message sources distributed among 500-1000 hosts. During development and testing, it should be possible to assemble pieces of the message logging system to provide all needed functionality on a much smaller scale. For example, when running tests on a single DCB board, if the DCB board provides access to local or remotely-mounted disk, then it should be possible to log messages to a file without starting a message server/concentrator on a separate host.

3.3 Client interface

Client libraries should be provided to allow the straightforward addition of message logging to application code.

3.3.1 API calls should be non-blocking

Calls to error logging routines from client applications should not block independent of the state of the message logging system. In cases in which the message can not be immediately delivered, it may be saved until such time as it can be delivered, or it may be discarded if sufficient buffer space is not available.

3.3.2 Supported languages

C, C++, Java, Python

3.4 Servers/concentrators

The system should provide applications that can serve as central receivers of messages for the purpose of concentrating messages in one location and providing convenient sources of messages for higher level elements in the system.

3.5 Message destinations

Various types of logging destinations must be supported, and each process in the system should be able to specify that its messages be routed to one or more of the possible destinations.

3.5.1 *File destinations*

Message logging to a file should be supported for all message sources. It should be possible to configure log files to maintain a fixed size and overwrite old entries with newer ones (a rolling log) or to keep all messages but periodically close the log file and start a new file when a certain time has passed or a certain file size is reached.

3.5.2 *Secondary process destinations*

It should be possible to send messages to a server process like the ones described above. The server process may be running on the same host as the source process or a different host.

3.5.3 *Database destinations*

Do we want to support logging to a local database?

3.6 Message filtering

The system should support filtering of messages at various levels.

3.6.1 *At the display level*

3.6.2 *At the sending level*

3.6.3 *Throttling of messages (Smoothing of bursts)*

The system should allow user-configurable suppression of message bursts. In cases in which messages are suppressed, the system should report the occurrence to higher-level entities so that end users know that the suppression occurred.

3.7 Message definition

Each message should contain the following information:

- Text to be displayed to the user or developer
- Type information that can be used to classify messages (error severity or debug level)
- An alphanumeric message code that can contain encoded information about the message type, text, and generating application.
- Information about the source of the message, including the application name that generated the message, the process ID of the application, the

- function or method name (and line number) that generated the message, and the name of the host on which the application is running.
- The time at which the message was created
 - The time at which the message was received by any intermediate servers or concentrators.

The elements of a message should be defined by the message logging system and built into clients that use the system, but the contents should not need to be compiled into the system. In that way, new messages may be added without needing to change every part of the message logging system.

3.8 Interface(s) to third party tools

As much as possible, it should be easy to extend the system to interface with third-party tools such as open-source log viewers.

3.9 Message-based actions

3.10 Remote display

The system should support the display of messages at remote locations.

3.11 Configuration

The system should support configuration of its behavior by users and system experts. The features that should be configurable include the following:

- For each message source, the destination(s) for messages. Multiple destinations should be allowed, and it should be possible to specify that different message types will be routed to different destinations.
- For each message source, the amount of message filtering that should be applied. For message generators, a filter may translate to a blocking of the generation of certain messages. For message servers or concentrators, a filter will prevent certain messages from being forwarded.
- The amount of buffering (if any) for various sources.
- Characteristics of log files (maximum size, cycle frequency, etc.)
- Throttling parameters
- Actions that are taken in response to specified messages

3.11.1 *Remote Configuration*

It should be possible to reconfigure message sources, concentrators, displays, etc. remotely (programmatically).

3.11.2 *Persisting configurations*

When the configuration of an element in the message system changes, do we want to save it somehow so that if the element restarts, the new configuration is used when the element restarts?

3.11.3 *Dynamic reconfiguration*

Reconfiguration should be possible at any time, not just between data taking runs.

4 Testing – QA/QC

The system should provide test suites for verifying its operation