The ACS Logging System is comprised of several pieces of software available for different programming languages. There are 4 main pieces or concepts regarding the logging system:

- Logging technology
- Logging implementation
- Notification Channel
- Central Logger
- *ACS Log Service

* The ACS Log Service is only used by the Python logging implementation, but it could be used by any ACS / CORBA application.

Logging Technology

The logging technology is the underlying library, package or framework that is used to enable the ACS logging system. Each programming language is based and/or compatible with at least one logging technology. The current support is the following:

- Python
  - Logging facility for Python
- Java
  - JDK Logger
  - SLF4J
  - Log4j2
- C++
  - ACE Logging
  - ~~Log4CPP~~

Each of these technologies is integrated with different ad-hoc techniques intrinsic to the mechanics and best practices they have, but also due to existing restrictions at the time of design and implementation that may no longer be there.

Logging Implementation

The logging implementation is the programming language specific code to integrate one or more of the logging technologies into ACS. The code for each of the programming languages is in different modules:

- Python: ACS/LGPL/CommonSoftware/acspycommon
- Java: ACS/LGPL/CommonSoftware/acsjlog
- C++: ACS/LGPL/CommonSoftware/logging

Python

The Python logging implementation is based in the standard Python 'logging' module. Some conventions are introduced to standardize log levels among the implementations as well as other convenient functionalities. The logging.Logger class is specialized for ACS by defining a local and a remote handler.

The local handler just formats the logs in a convenient way to stdout, while the central handler takes advantage of the ACS Log Service to distribute the logs to any interested client.

Java

C++

The C++ logging implementation is based in the ACE Logging framework which imposes several restrictions and complications for the implementation. The design for the logging system is similar to log4j, python logging, etc. which is convenient and familiar, offering different handlers for stdout, file, central logger, etc.

As opposed to Java and Python, in C++ it was decided that the logger reference would not be handled by developers, but instead there are macros that can be called from any place and those macros have the responsibility of obtaining an appropriate logger instance for performing the log calls.

Notification Channel

The Notification Channel is a CORBA service for decoupled messaging through events. We're currently using the implementation provided by ACE/TAO through the Notify Service. For more details please look at the Notification Channel technical document.

Central Logger

The Central Logger is a CORBA-based service that is used to receive a set of logs and redistribute it to interested parties in a decoupled way through the Notification Channel.

This service is implemented in C++ and is part of the ACS/LGPL/CommonSoftware/logging module.

ACS Log Service

The ACS Log Service is a simple CORBA service that receives log requests to be propagated to the Central Logger. It takes advantage of the C++ logging implementation and simply calls the existing C++ macros that will call the Central Logger service methods.

This service is implemented in C++ and is part of the ACS/LGPL/CommonSoftware/acslog module.

- ACS Technical Documents - Logging System - C++ ACE Logger Implementation
- ACS Technical Documents - Logging System - C++ Boost Log Implementation