OpenClovis
Software Development Kit (SDK)
Service Description and API Reference for Log Service

For OpenClovis SDK Release2.3 V0.4

Document Revision Date: December 22, 2006

Contents

Chapter 1

Functional Overview

The OpenClovis Log Service collects, translates, and publishes log messages to record any significant event in the system. For instance, operational state change of a component, managed object attribute value change, and so on. Log analyzes the proper system behavior and registers the potential unintended operations. Log-levels can be changed during the operation. The log service supports syslog and SAF-defined logging facility.

The Log Service provides the following functions:

- Logs a message
- Opens an application stream
- Closes an application stream
- Sets the log level of a component
- Returns the log level of a component
- Sets the system log level (either of a single node or the entire cluster).
- Verifies the version used by the invoking process.

  Before invoking any of the functions, it is necessary to invoke *clLogLibInitialize()* function after which an EO can either open its own private stream to log the message or use the default stream to log. Two default streams are provided i.e., `CL_LOG_STREAM_APP` and `CL_LOG_STREAM_SYS`. These two streams ideally lead to different files where the logs will be present. It is possible to create only application streams (system streams cannot be created).

The log library interacts with the following components:

- Buffer library for maintaining its circular buffer.
- OSAL library for allocating and de-allocating its circular buffer.
- RMD for communicating with the Log Server.

Chapter 2

Service APIs

2.1 Type Definitions

2.1.1 ClLogCloseWrpr

*typedef ClRcT(*ClLogCloseWrpr)(ClLogStreamHdlT handle)*

The type of the handle supplied by the Log Service that closes a stream function pointer.

2.1.2 ClLogFinalizeWrpr

typedef ClRcT(ClLogFinalizeWrpr)()

This finalizes a function pointer.

2.1.3 ClLogOpenWrpr

*typedef ClRcT(ClLogOpenWrpr)(ClLogFormatFileConfigT fileConfig, ClLogStreamHdlT *handle)*

The type of the handle supplied by the Log Service that opens a stream function pointer.

2.1.4 ClLogStreamHdlT

typedef ClHandleT ClLogStreamHdlT

The type of the handle supplied by the Log Service stream.

2.1.5 ClLogWriteWrpr

typedef ClRcT (ClLogWriteWrpr)(ClLogHeaderT)

This is a Write function pointer.

2.1.6 ClLogFileConfigT

typedef struct {

textitClCharT fileLoc [CL_LOG_FILEPATH_LENGTH];

textitClCharT fileName [CL_LOG_FILENAME_LENGTH];

textitClUint64T maxFileSize;

textitClUint32T maxRecordSize;
} ClLogFileConfigT;

The structure *ClLogFileConfigT* contains the configuration information of the Log file.

- *fileLoc* - The location of the Log file.
- *fileName* - Original file name.
- *maxFileSize* - Maximum size of file.
- *maxRecordSize* - Maximum log size.

2.2 Library Life Cycle Functions

2.2.1 clLogLibInitialize

clLogLibInitialize

Synopsis:
Initializes the Log client library.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogLibInitialize(void);
```

Parameters:
None.

Return values:
CL_OK: The API executed successfully.

CL_ERR_INITIALIZED: The Log Service is already initialized.

Description:
This function initializes the Log Service for the invoking process. No callbacks are registered. It initializes the ring buffer that is used to store the logs, when Log Server is down. It also initializes all mutexes used in the code and starts a new thread that is used for flushing the Ring Buffers. This function must be invoked prior to the invocation of any other Log Service functionality.

Library File:
ClLogClient

Related Function(s):
clLogLibFinalize

2.2.2 clLogLibFinalize

clLogLibFinalize

Synopsis:
Cleans up the Log library.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogLibFinalize(void);
```

Parameters:
None

Return values:
CL_OK: The API executed successfully.

Description:
This function is used to clean up the Log client library. On successful execution of this function, it releases all the resources allocated during the initialization of the library.

Library File:
ClLogClient

Related Function(s):
clLogLibInitialize

2.3 Functional APIs

2.3.1 clLogSystemLevelSet

clLogSystemLevelSet

Synopsis:
Sets the System log level.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogSystemLevelSet(
                            CL_IN  ClLogSeverityT  severity,
                            CL_IN  ClIocNodeAddressT   nodeAddress,
                            CL_IN  ClLogFilterPatternT pattern);
```

Parameters:

Severity : The log level to be set.

NodeAddress: The node at which the log level is to be set. Either a specific node address can be given or a broadcast address (`CL_IOC_BROADCAST_ADDRESS`)can be given to set the log level of all nodes. using the `CL_IOC_BCAST_ADDR`.

Pattern: Specifies the SG, SU and component, to which the filter is applied.

Return values:

CL_OK: The API executed successfully.

CL_ERR_TIMEOUT: When the Log Server is down.

CL_ERR_NOT_INITIALIZED: The log library is not initialized.

CL_ERR_TRY_AGAIN: The Log Server is up but is not operational.

CL_ERR_VERSION_MISMATCH: The version of log library is not supported.

Description:
This function is used to set the system log level. An invocation of *clLogSystemLevelSet()* is non-blocking with no reply. The severity can be set on two basis i.e, either on a single node or all node. For all nodes, use `CL_IOC_BCAST_ADDR`.

Library File:
ClLogClient

Related Function(s):
clLogLevelSet , clLogLevelGet

2.3.2 clLogWrite

clLogWrite

Synopsis:
Writes the Log record.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogWrite(
                            CL_IN ClLogStreamHdlT streamHdl,
                            CL_IN ClLogSeverityT severity,
                            CL_IN ClCharT *libName,
                            CL_IN ClCharT *msg,
                            ...);
```

Parameters:

streamHdl: Handle to the stream. It must be one of the default handles or the one obtained through *clLogOpen()*.

severity: Severity of the message.

libName: Name of the library being logged. It must be NULL only when the record comes from the server.

msg: A non-NULL pointer to the log message.

vargs: Variable number of arguments that is formatted with the log message.

Return values:

CL_OK: The API executed successfully.

CL_ERR_NOT_INITIALIZED: The log library is not initialized.

CL_LOG_ERR_UNDEFINED_SEVERITY: The severity is out of bounds.

CL_ERR_NULL_POINTER: The log message is NULL.

Description:
This function is used to log a record to a stream specified by the *streamHdl*. The *streamHdl* is either the default handles provided or the one obtained from *clLogOpen()*. An invocation of *clLogWrite()* is non-blocking. The callback is provided with the RMD call and the return value is not returned to the invoking process.

Library File:
ClLogClient

Related Function(s):
clLogOpen

2.3.3 clLogLevelSet

clLogLevelSet

Synopsis:
Sets the Log level of the Logger.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogLevelSet(
                            CL_IN ClLogSeverityT severity);
```

Parameters:
severity: Severity to be set.

Return values:
CL_OK: The API executed successfully.

CL_LOG_ERR_UNDEFINED_SEVERITY: When the severity is out of bounds.

CL_ERR_NOT_INITIALIZED: The log library is not initialized.

Description:
This function is used to set the log level of the invoking process. This call is local to the Log Client. An invocation of *clLogLevelSet()* is blocking. At the time of initialization of the log library, by default the log level of the invoking process is set to `CL_LOG_ERROR`. The *clLogLevelSet()* function can be used to set the log level to the required severity.

Library File:
ClLogClient

Related Function(s):
clLogLevelGet , clLogSystemLevelSet

2.3.4 clLogLevelGet

clLogLevelGet

Synopsis:
Returns the current Log level of the Logger.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogLevelGet(
                            CL_OUT ClLogSeverityT *severity);
```

Parameters:
severity: (out) Log level of the application.

Return values:
CL_OK: The API executed successfully.

CL_ERR_NOT_INITIALIZED: The log library is not initialized.

Description:
This function is used to return the current Log level of the invoking process of which the client is a part. The application may need to know its current log level at any point of time. It can use this level to log messages. There are no pre-requisites, except that the library must be initialized prior to calling this function. This call is local to the log library. An invocation of *clLogLevelGet()* is blocking.

Library File:
ClLogClient

Related Function(s):
clLogLevelSet, clLogSystemLevelSet

2.3.5 clLogOpen

clLogOpen

Synopsis:
Opens a stream by any Application Logger.

Syntax:

```
ClRcT clLogOpen(
                          CL_OUT ClLogStreamHdlT *handle,
                          CL_IN  ClLogFormatFileConfigT    fileConfig);
```

Parameters:

handle: (out) A non-NULL handle to the application stream. This handle is used subsequently in the *clLogWrite()* function to log messages.

fileConfig: The file attributes of the stream. The format in the *fileConfig* should be one of the characters 'C/c', 'P/p', 'S/s', 'I/i', 'M/m'. Maximum of five characters are permitted, the excess characters are ignored.

Return values:

CL_OK: The API executed successfully.

CL_ERR_NOT_INITIALIZED: The log library is not initialized.

CL_ERR_TRY_AGAIN: The Log Server is up but not operational.

CL_LOG_ERR_OPEN_NOT_SUPPORTED: This API is not supported with this mode.

CL_ERR_VERSION_MISMATCH: The version of log library not supported.

CL_LOG_ERR_INVALID_FORMAT_EXPR: The format expression is invalid.

CL_LOG_INTERNAL_ERROR: An internal error occurred at the server.

Description:
This function is used to create and open a stream by an invoking process. Loggers can use this newly returned handle or the default handles provided to log their messages. Only application streams can be created by the invoking process. An invocation of *clLogOpen()* is blocking.

Library File:
ClLogClient

Related Function(s):
clLogClose , clLogWrite

2.3.6 clLogClose

clLogClose

Synopsis:
Closes a stream opened by the Application Logger.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogClose(
                            CL_IN ClLogStreamHdlT handle);
```

Parameters:
handle: (out) Handle of the stream returned from *clLogOpen()*.

Return values:
CL_OK: The API executed successfully.

CL_ERR_NOT_INITIALIZED: The log library is not initialized.

CL_ERR_TRY_AGAIN: The Log Server is up but not operational.

CL_LOG_ERR_CLOSE_NOT_SUPPORTED: This API is not supported with this mode.

CL_ERR_VERSION_MISMATCH: The version of log library not supported.

Description:
This function is used to close a stream opened by the application logger. The handle is returned by *clLogOpen()*. An invocation of *clLogClose()* is blocking. Subsequent invocation of *clLogWrite()* function with this handle will fail.

Library File:
ClLogClient

Related Function(s):
clLogWrite, clLogOpen

2.3.7 clLogVersionVerify

clLogVersionVerify

Synopsis:
Verifies the version information.

Header File:
clLogApi.h

Syntax:

```
ClRcT clLogVersionVerify(
                            CL_INOUT   ClVersionT  *pVersion);
```

Parameters:

pVersion: (in/out) The version information. Current version supported is 'B', 0x1. 0x1. As input parameter, the application can pass the version of the log client it requires to query for, to check whether the client supports it. If the version is not supported by the client, then `CL_VERSION_MISMATCH` is returned, otherwise the nearest version which the client can support is returned as output parameter.

Return values:

CL_OK : If the client library supports the version

CL_VERSION_MISMATCH: If the client library does not support the version specified by the invoking process.

Description:
This function is used to verify if a version of the log library is supported. This function must be invoked after initialization to check whether the *Eoversion* is supported by client.

Library File:
ClLogClient

Related Function(s):
None.

Index