NAME

stli – Standard Tape Library Interface

DESCRIPTION

STLI is the documented interface for connecting tape libraries to NetWorker. Tape libraries are often referred to as silos. Examples of tape libraries are StorageTek ACSLS silos, ABBA systems from Grau or the IBM 3494. Tape libraries differ from other jukeboxes supported by NetWorker in two major points:

- The tape libraries have their own volume and slot management. The volumes are accessed by NetWorker via their barcode labels. The exact location of a given tape is unknown to NetWorker.
- The interface to the library for issuing requests like mount or unmount of particular volumes differs from that of jukeboxes, which are normally connected via a SCSI or V24 interface. Standard Tape Library commands are transmitted to a Standard Tape Library server, which receives these commands and invokes the appropriate library actions. Usually, the connection between NetWorker and the Library controller is over the network, although serial (RS-232) connections are also used. Note that the actual connection is hidden from NetWorker by the STL library. Accordingly NetWorker and other applications calling the STL server are called library clients.

To keep applications independent of differing interfaces to various tape library systems, an API called STLI (Standard Tape Library Interface) has been defined, which is used by NetWorker to invoke library requests. The STLI specifies a shared library with well defined functions, which is dynamically linked to NetWorker. These STLI interface libraries, which transform the STLI function calls to library-specific calls to the proprietary tape library server, may be provided by Legato or the manufacturers of the tape library.

Not all functions specified in this paper must be implemented in the STLI library. These functions are the minimum necessary for a functional library:

```c
stl_open()
stl_close()
stl_mount()
stl_unmount()
```

Implementation of stl_error() is recommended for easier use of the library and better troubleshooting.

If you wish to support dynamic device reservation these are the relevant functions:

```c
stl_reserve_dev()
stl_release_dev()
stl_dev_reservation()
```

Optional functions for added features:

```c
stl_query_volume()
stl_withdraw_volume()
stl_withdraw_volumes()
stl_deposit_volume()
stl_deposit_volumes()
stl_version()
```

```c
stl_close()
```

Declaration:

```
int stl_close(char * stl);
```

Description:

Close the connection to the tape library.
<stl> is the handle returned by stl_open().

The return value on success is STL_ERR_NOERR. For error return values see appendix.

stl_deposit_volume

Declaration:
```c
int stl_deposit_volume(char *stl, char *volume, char *capname)
```

Description:
Causes the specified volume to be inserted into the library from the specified cartridge access port (inport/export facility, mailslot....)

<stl> is the handle returned by stl_open().
<volume> is the barcode of the volume to be deposited into the library.
<capname> specifies the cartridge access port/inport-export area/mailslot to be used to insert the volume into the library. It is a character string, which is understood by the tape library as a name for that device. This argument can be NULL, in which case the default CAP will be used.

This function returns STL_ERR_NOERR if the volume is successfully inserted. STL_ERR_NOVOL is returned if the volume is not present and was not inserted. Other return values are possible if errors occur. See appendix for possible values.

NOTE: on some libraries, this function may not be needed.

- The IBM 3494 will automatically import any tapes placed into its in/out area. There is no 'deposit' function in the 3494’s API.
- StorageTek libraries with the CAP set to automatic mode behave in the same manner as the 3494. However, if their CAP is set to manual, then a deposit call is required.
- On EMASS/Grau libraries, a single call to deposit one volume from the EIF will in fact deposit all available volumes. However, subsequent deposit calls will return quickly and the error can be ignored.

stl_deposit_volumes

Declaration:
```c
int stl_deposit_volumes(char *stl, char *volumes, char *capname)
```

Description:
Causes the specified volumes to be inserted into the library from the specified cartridge access port (inport/export facility, mailslot....). This function will be used instead of stl_deposit_volume() if it is defined and stl_version returns 1.3 or greater. Therefore, it should be capable of functioning with either a single volume specified or with a comma separated list.

<stl> is the handle returned by stl_open().
<volumes> is a comma separated list of barcodes of the volumes to be deposited into the library. There should be no extraneous spaces added between the individual barcodes since the space character (ASCII 32) is a valid barcode character itself.
<capname> specifies the cartridge access port/inport-export area/mailslot to be used to insert the volume into the library. It is a character string, which is understood by the tape library as a name for that device. This argument can be NULL, in which case the default CAP will be used.

This function returns STL_ERR_NOERR if the volume is successfully inserted. STL_ERR_NOVOL is returned if the volume is not present and was not inserted. Other return values are possible if errors occur. See appendix for possible values.

NOTE: on some libraries, this function may not be needed.
o The IBM 3494 will automatically import any tapes placed into its in/out area. There is no ‘deposit’ function in the 3494’s API.

o StorageTek libraries with the CAP set to automatic mode behave in the same manner as the 3494. However, if their CAP is set to manual, then a deposit call is required.

o On EMASS/Grau libraries, a single call to deposit one volume from the EIF will in fact deposit all available volumes. However, subsequent deposit calls will return quickly and the error can be ignored.

stl_dev_reservation()

Declaration:

    int stl_dev_reservation(char *stl, char *device, int *state)

Description:

    Get the reservation state of device <device>.
    <stl> is the handle returned by stl_open().
    <device> specifies the device to get the reservation state of. It is a character string, which is understood by the tape library as a name for that device.
    * <state> returns the reservation state:
      STL_DEV_FREE: Free
      STL_DEV_RESERVED: Reserved for NetWorker’s use
      STL_DEV_OCCUPIED: Occupied by another host
    The return value on success is STL_ERR_NOERR. For error return values see appendix.

stl_error()

Declaration:

    char *stl_error(void)

Description:

    Gives a printable error message belonging to the preceding STLI function call. The function returns the address of a buffer which contains a message describing the status of the last STLI function call. These messages can be constant strings, for instance the messages contained in stl.h for error codes less than 100. But these messages can also be built up with actual parameters, which describe more exactly the error situation.
    Error and other status information should be maintained in global static variables in the library, since this call is made without any parameters.
    The function returns NULL if no message available. See appendix.

stl_mount()

Declaration:

    int stl_mount(char* stl, char* volume, char* device);

Description:

    Move volume <volume> into drive <drive>.
    <stl> is the handle returned by stl_open().
<volume> is the barcode of the volume to be mounted.
<device> specifies the device, on which the volume shall be mounted. It is a character string, which is understood by the tape library as a name for that device.
This call may not return until the volume is loaded into the drive and the drive has come ready. The exact sequence is dependent on the library. It can therefore take several minutes to complete.
The return value on success is STL_ERR_NOERR. For error return values see appendix.

stl_open()
Declaration:
int stl_open(char * server, char ** stl);
Description:
Connect to the tape library.
<server> is a character string which contains all information necessary to establish a connection to the tape library. The information in this string is proprietary to the special type of tape library. Generally it should be of the form:
[<host>] [ <par1>=<val1> [ <par2>=<val2> ] ... ]
In most cases the string contains merely <host>, the nodename of a library server, which receives and serves the STLI requests over the network.
*<stl> is a handle returned for use by the other STLI functions. This handle can be used to store internal information between subsequent function calls. For some libraries, the parameter to this call may or may not be used, as environment variables may be used to hold the required configuration information.
The return value on success is STL_ERR_NOERR. For error return values see appendix

stl_query_volume
Declaration:
int stl_query_volume(char *stl, char *volume)
Description:
Queries a silo to establish the presence of a volume. This function is currently used to verify the presence of a volume before allocating that volume for use with NetWorker.
<stl> is the handle returned by stl_open().
<volume> is the barcode of the volume to be mounted.
This function returns STL_ERR_NOERR if the volume is present, or STL_ERR_NOVOL if the volume is not present.
Other return values are possible if errors occur. See appendix for possible values.

stl_release_dev()
Declaration:
int stl_release_dev(char *stl, char *device);
Description:
Release device <device>, which has previously been reserved by stl_reserve_dev().
<stl> is the handle returned by stl_open().
<device> specifies the device to be released. It is a character string, which is understood by the
tape library as a name for that device.

The return value on success is STL_ERR_NOERR. For error return values see appendix.

stl_reserve_dev()
Declaration:
    int stl_reserve_dev(char *stl, char *device);
Description:
    Reserves device <device> for NetWorker’s use.
    <stl> is the handle returned by stl_open().
    <device> specifies the device to be reserved. It is a character string, which is understood by the
tape library as a name for that device.
    The return value on success is STL_ERR_NOERR. For error return values see appendix.

stl_version()
Declaration:
    int stl_version(void)
Description:
    Returns STLI version information for the STL library
    This function returns the version of the STL library * 10. I.e., it returns a value of 12 for an STL
library that supports the functions for STLI version 1.2.

STLI version 1.0 specified the following calls:
    stl_close()
    stl_dev_reservation()
    stl_mount()
    stl_open()
    stl_release_dev()
    stl_reserve_dev()
    stl_unmount()

STLI version 1.1 added the following calls:
    stl_query_volume()
    stl_version()

STLI version 1.2 added the following calls:
    stl_deposit_volume()
    stl_withdraw_volume()

STLI version 1.3 added the following calls:
    stl_deposit_volumes()
    stl_withdraw_volumes()
Note that stl_version does not return a value that can be interpreted as an ’STL_’ error. Attempting
to do so will result in unpredictable results.

stl_unmount()
Declaration:

    int stl_unmount(char * stl, char * volume, char * device);

Description:

    Remove volume <volume> from drive <drive>.
    <stl> is the handle returned by stl_open().
    <volume> is the barcode of the volume to be removed.
    <device> specifies the device, from which the volume shall be removed. It is a character string,
    which is understood by the tape library as a name for that device.
    Either <volume> or <device> can be NULL. If both values are specified, they must be consistent.
    This call will not return until the volume is ejected from the drive and returned to its slot by the
    library. It can therefore take several minutes to complete.
    The return value on success is STL_ERR_NOERR. For error return values see appendix.

stl_withdraw_volume

Declaration:

    int stl_withdraw_volume(char *stl, char *volume, char *capname)

Description:

    Causes the specified volume to be ejected from the library through the specified cartridge access
    port (inport/export facility, mailslot....)
    <stl> is the handle returned by stl_open().
    <volume> is the barcode of the volume to be withdrawn.
    <capname> specifies the cartridge access port/inport-export area/mailslot to be used to remove the
    volume from the silo. It is a character string, which is understood by the tape library as a name for
    that device. This argument can be NULL, in which case the default CAP will be used.
    This function returns STL_ERR_NOERR if the volume is successfully withdrawn from the
    library. STL_ERR_NOVOL is returned if the volume is not present and STL_ERR_VOLBUSY is
    returned if the volume is currently in use and cannot be withdrawn.
    Other return values are possible if errors occur. See appendix for possible values.

stl_withdraw_volumes

Declaration:

    int stl_withdraw_volumes(char *stl, char *volumes, char *capname)

Description:

    Causes the specified volume to be ejected from the library through the specified cartridge access
    port (inport/export facility, mailslot....) This function will be used instead of stl_withdraw_volume() if it is defined and stl_version returns 1.3 or greater. Therefore, it should be capable of
    functioning with either a single volume specified or with a comma separated list.
    <stl> is the handle returned by stl_open().
    <volumes> is a comma separated list of barcodes of the volumes to be deposited into the library.
    There should be no extraneous spaces added between the individual barcodes since the space char-
    acter (ASCII 32) is a valid barcode character itself.
    <capname> specifies the cartridge access port/inport-export area/mailslot to be used to remove the
    volume from the silo. It is a character string, which is understood by the tape library as a name for
that device. This argument can be NULL, in which case the default CAP will be used.

This function returns STL_ERR_NOERR if the volume is successfully withdrawn from the
library. STL_ERR_NOVOL is returned if the volume is not present and STL_ERR_VOLBUSY is
returned if the volume is currently in use and cannot be withdrawn.

Other return values are possible if errors occur. See appendix for possible values.

Appendix: Return Values

Return values 0 - 99 are reserved for common, library type independent error codes. The header file stl.h
defines the common return values together with a short error message.

Return values greater 100 can be used by each STLI implementation for proprietary error codes.

It is recommended, that a STLI implementation should map all error situations to the common STLI error
codes and should provide the function stl_error() for more detailed error messages. This allows NetWorker
to react on known error codes, but also to forward the more detailed error messages via the user interface.

No proprietary error codes are allowed in situations, where the common error codes
STL_ERR_DEVEMPTY, STL_ERR_DEVFULL or STL_ERR_ALRDYMNTED apply.

The currently defined return codes are:

<table>
<thead>
<tr>
<th>Error value</th>
<th>Meaning</th>
</tr>
</thead>
<tbody>
<tr>
<td>STL_ERR_NOERR</td>
<td>Successful call return - no error</td>
</tr>
<tr>
<td>STL_ERR_UNKNOWN</td>
<td>Error, no details available</td>
</tr>
<tr>
<td>STL_ERR_CONNECT</td>
<td>Cannot connect to tape library</td>
</tr>
<tr>
<td>STL_ERR_BUSY</td>
<td>Tape library busy, try later</td>
</tr>
<tr>
<td>STL_ERR_ACCESS</td>
<td>Permission denied (to access the library, the requested device, volume or operation)</td>
</tr>
<tr>
<td>STL_ERR_NODEV</td>
<td>Device not known to the tape library or physically not available</td>
</tr>
<tr>
<td>STL_ERR_NOVOL</td>
<td>Volume not known to the tape library or physically not available</td>
</tr>
<tr>
<td>STL_ERR_DEVFULL</td>
<td>Device already loaded with another volume</td>
</tr>
<tr>
<td>STL_ERR_DEVEMPTY</td>
<td>Device empty</td>
</tr>
<tr>
<td>STL_ERR_DEVBUSY</td>
<td>Device busy</td>
</tr>
<tr>
<td>STL_ERR_ERRNO</td>
<td>Local UNIX error, see errno</td>
</tr>
<tr>
<td>STL_ERR_INVALID</td>
<td>Invalid parameter</td>
</tr>
<tr>
<td>STL_ERR_VOLBUSY</td>
<td>Volume already loaded in another drive or is otherwise occupied</td>
</tr>
</tbody>
</table>
STL_ERR_LIBRARY
  Tape library internal error

STL_ERR_CONFIG
  Request doesn't comply with tape library configuration

STL_ERR_DEVOCC
  Device reserved by another host

STL_ERR_DEVRES
  Device already reserved

STL_ERR_DEVNOTRES
  Device not reserved

STL_ERR_NOTINST
  STL-Library is a dummy library

STL_ERR_NOTSUPP
  Dummy function return

STL_ERR_ALRDYMNTEd
  Requested volume already mounted in requested device

SEE ALSO
  nsrjb(8), nsr_jukebox(5), IBM_silo(8), EMASS_silo(8), STK_silo(8).