

TPOPEN(3)

| REVISION HISTORY | | | |
|------------------|------|-------------|------|
| NUMBER | DATE | DESCRIPTION | NAME |
| | | | |

Contents

| | | |
|----------|---------------------|----------|
| 1 | SYNOPSIS | 1 |
| 2 | DESCRIPTION | 2 |
| 3 | RETURN VALUE | 3 |
| 4 | ERRORS | 4 |
| 5 | EXAMPLE | 5 |
| 6 | BUGS | 6 |
| 7 | SEE ALSO | 7 |
| 8 | COPYING | 8 |

Chapter 1

SYNOPSIS

```
#include <atmi.h>
```

```
int tpopen (void);
```

For XATMI client link with *-latmiclt -latmi -lubf -lnstd -lpthread -lrt -lm*

For XATMI server link with *-latmisrvl -latmisrvnomainl -latmisrvinteg -latmi -lubf -lnstd -lpthread -lrt -lm*

Chapter 2

DESCRIPTION

Connect to the XA resource manager. Note that environment for process must be configured, so that Enduro/X can load correct resource manager drivers. See the **ex_env(5)** man page for environment variables, that must be set. Basically Enduro/X will dynamically load library set in **NDRX_XA_DRIVERLIB** env variable. This is the driver which provides in unified way of getting **xa_switch**. As the xa_switches can be located in different Resource Managers, the second library might be need to load and that is marked in **NDRX_XA_RMLIB** env variable. The resource manager ID is set in **NDRX_XA_RES_ID** env variable. If your application needs different resource managers or different transaction branches, then you can use environment variable override functionality, see **ex_envover(5)** and **ndrxconfig.xml(5)**. XA connection strings for Enduro/X drivers are stored into **NDRX_XA_OPEN_STR** and **NDRX_XA_CLOSE_STR** variables. Enduro/X will try to init XA sub-system as soon as possible (i.e. load drivers, etc.). If you want to delay this till actual **tpopen()**, then you may set **NDRX_XA_LAZY_INIT** to 1.

This function at lowest level, basically calls XA switches **xa_open_entry()** function with **NDRX_XA_OPEN_STR** string.

If **tpopen()** was already call and XA is initialised, then function will succeed, and no error will be generated.

Chapter 3

RETURN VALUE

On success, **tpopen()** return zero; on error, -1 is returned, with **tperrno** set to indicate the error.

Chapter 4

ERRORS

Note that **tpsterror()** returns generic error message plus custom message with debug info from last function call.

TPERMERR Resource Manager failed. The **tpsterror()** will provide more info from last call.

TPESYSTEM System failure occurred during serving. See logs i.e. user log, or debugs for more info. This could also be a problem with dynamical driver loading.

TPEOS System failure occurred during serving. See logs i.e. user log, or debugs for more info.

Chapter 5

EXAMPLE

See `atmitest/test021_xafull/atmict21.c` for sample code.

Chapter 6

BUGS

Report bugs to madars.vitolins@gmail.com

Chapter 7

SEE ALSO

`tpcommit(3)` `tpopen(3)` `tpclose(3)` `tpsuspend(3)` `tpresume(3)`

Chapter 8

COPYING

© Mavimax, Ltd