

**TPCALL(3)**

REVISION HISTORY			
NUMBER	DATE	DESCRIPTION	NAME

# Contents

<b>1</b>	<b>SYNOPSIS</b>	<b>1</b>
<b>2</b>	<b>DESCRIPTION</b>	<b>2</b>
<b>3</b>	<b>RETURN VALUE</b>	<b>3</b>
<b>4</b>	<b>ERRORS</b>	<b>4</b>
<b>5</b>	<b>EXAMPLE</b>	<b>5</b>
<b>6</b>	<b>BUGS</b>	<b>6</b>
<b>7</b>	<b>SEE ALSO</b>	<b>7</b>
<b>8</b>	<b>COPYING</b>	<b>8</b>

## Chapter 1

# SYNOPSIS

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

```
int tpcall(char *svc, char *idata, long ilen, char **odata, long *olen, long flags);
```

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

For XATMI server link with *-latmisrv|-latmisrvnomain|-latmisrvinteg -latmi -lubf -lnstd -lpthread -lrt -lm*

## Chapter 2

# DESCRIPTION

Call the XATMI service by given *svc* name. The *idata* is optional input XATMI buffer. If it is not a NULL, then it must be allocated with **tpccalloc()** call, *ilen* is used for buffer types such **CARRAY**, where buffer length is not described by type. When caller receives response, it is stored in *odata* buffer which must be also allocated by **tpccalloc()**. If the allocated buffer size is shorter than received one, then Enduro/X will automatically realloc the buffer to new size. *olen* is mandatory field, where the received buffer length is stored. If **TPNOTRAN** is not specified and current process is in global transaction, then system will make destination process run in same destination process.

### Valid flags

**TPNOTRAN** Do not call service in transaction mode. This is effective in case if caller process is running in transaction mode, but destination process shall not run in the same global transaction

**TPSIGRSTRT** Restart the system call in progress if interrupted by signal handler. This affects only underlaying mq\_\* function calls.

**TPNOTIME** Ignore timeout setting (**NDRX\_TOUT** env variable). Wait for reply for infinitely.

**TPNOCHANGE** Do not allow to change the reply buffer type. If flag is set and different buffer type is received than original, then error **TPEINVAL** is returned.

**TPTRANSUSPEND** Suspend the current transaction in progress and continue it with destination process. This is suitable in cases when **XA** adapter does not allow multiple processes/sessions to have active same transaction in the same transaction branch.

**TPNOBLOCK** In case of target service request queue is full, do not wait on queue, but return error. The error code for this situation is **TPEBLOCK**. This affects only request part of the call. This flag does not affect waiting for response from server.

## Chapter 3

# RETURN VALUE

On success, **tpcall()** returns 0; on error, -1 is returned, with **tperrno** set to indicate the error. When server did **tpreturn()**, the *rcode* value is accessible by caller by using **tpurcode()**.

## Chapter 4

# ERRORS

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

**TPEINVAL** Invalid parameter is given to function. Either service name is NULL.

**TPEOTYPE** Flags does not allow to change the response buffer type.

**TPENOENT** No service (*svc* parameter) advertised in system.

**TPETIME** Service did not reply in given time (*NDRX\_TOUT*).

**TPESVCFAIL** Service returned *TPFAIL*. This is application level failure.

**TPESVCERR** System level service failure. Server died during the message presence in service queue.

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

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

**TPEBLOCK** Service request queue was full and **TPNOBLOCK** flag was specified.

**TPNOABORT** Do not abort global transaction (if one in progress), even if service failed.

## Chapter 5

# EXAMPLE

See `atmitest/test001_basiccall/atmiclt1.c` for sample code.



## Chapter 6

# BUGS

Report bugs to [support@mavimax.com](mailto:support@mavimax.com)

## Chapter 7

## SEE ALSO

**tpacall(3) tpgetrply(3)**

## Chapter 8

# COPYING

© Mavimax, Ltd