

**TPBEGIN(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 tpbegin (unsigned long timeout, long flags);
```

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

Begin the global transaction. The current user must be open **XA** subsystem by **topopen()**. *timeout* is setting for transaction manager (**tmsrv**) how long transaction can hang in active state (i.e. not prepared/committed). Field *flags* is reserved for future use and must be 0. The work done by **tpbegin()** transaction is only in scope of processes which uses **XA** interfaces for resource managers. Any others works are outside of current scope. If *timeout* is set to 0, then default maximum time for transaction processing is used. Transaction must be committed or aborted by client process by using **tpcommit()** or **tpabort()**. If any **tpcall()** with no **TPNOTRAN** flag fails within given global transaction, then transaction is automatically marked as abort only.

Client process can suspend the global transaction by **tpsuspend(3)** and some other process can resume the transaction as client with suspend data. In that case that other process can do the commit.

## Chapter 3

# RETURN VALUE

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

## Chapter 4

# ERRORS

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

**TPEINVAL** flags was not 0.

**TPETIME** Transaction manager (**tmsrv(8)**) did not respond in configured time-out time. The state of transaction is unknown.

**TPESVCERR** Failed to call transaction manager, with service error. The state of transaction is unknown.

**TPEPROTO** XA subsystem was not initialized (did not call **tpopen()**) or global transaction already started.

**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.

## Chapter 5

# EXAMPLE

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



## Chapter 6

# BUGS

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

## Chapter 7

## SEE ALSO

**tpbegin(3) tpcommit(3) tpopen(3) tpclose(3) tpsuspend(3) tpresume(3)**

## Chapter 8

# COPYING

© Mavimax, Ltd