ToolTalk™ Setup and Administration Guide

Document Number 007-1617-020

ToolTalk™ Setup and Administration Guide
Document Number 007-1617-020

Contents

Figures

Tables

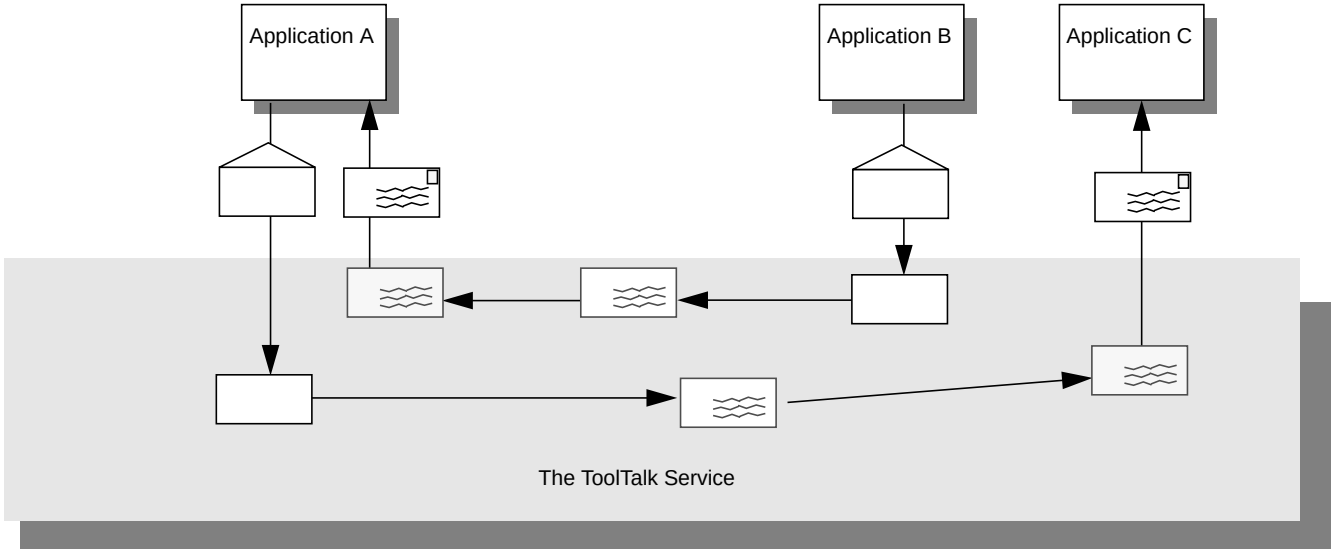
Chapter 1

The ToolTalk Service Overview

This chapter provides background information about the ToolTalk™ service. It also highlights the components that you, as a system administrator, should be familiar with in order to respond to user questions about ToolTalk messages. The remaining chapters of this manual contain detailed installation and maintenance instructions.

Introducing the ToolTalk Service

The ToolTalk service enables independent applications to communicate with each other without having direct knowledge of each other. Applications create and send ToolTalk messages to communicate with each other. The ToolTalk service receives these messages, determines the recipients, and then delivers the messages to the appropriate ToolTalk applications, as shown in Figure 1-1.



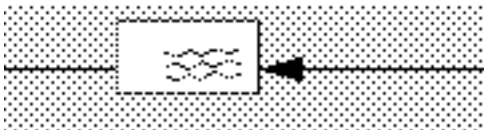


Figure 1-1 Applications Using the ToolTalk Service

To use the ToolTalk service to communicate, applications agree on a *message protocol*. A message protocol is a set of ToolTalk messages that describe operations the applications agree to perform. The message protocol specification includes the set of messages and specifies how applications should behave when they receive each message.

ToolTalk Architecture

The following ToolTalk service components work together to provide interapplication communication and object information management:

- `ttsession` is the ToolTalk communication process.

  One `ttsession` runs in an *X* server session or process tree session and communicates with other `ttsession`s when a message needs to be delivered to an application in another session.

- `rpc.ttdbserverd` is the ToolTalk database server process.

  One `rpc.ttdbserverd` is installed on each machine which contains a disk partition that stores files of interest to ToolTalk clients or files that contain ToolTalk objects.

  File and ToolTalk object information is stored in a records database managed by `rpc.ttdbserverd`.

- `libtt` is the ToolTalk application programming interface (API) library.

  Applications include the API library in their program and call the ToolTalk functions in the library.

The ToolTalk service uses Remote Procedure Call (RPC) to communicate between these ToolTalk components.

Applications provide the ToolTalk service with process and object type information. This information is stored in an XDR formatted types database.

Figure 1-2 illustrates the ToolTalk service architecture.

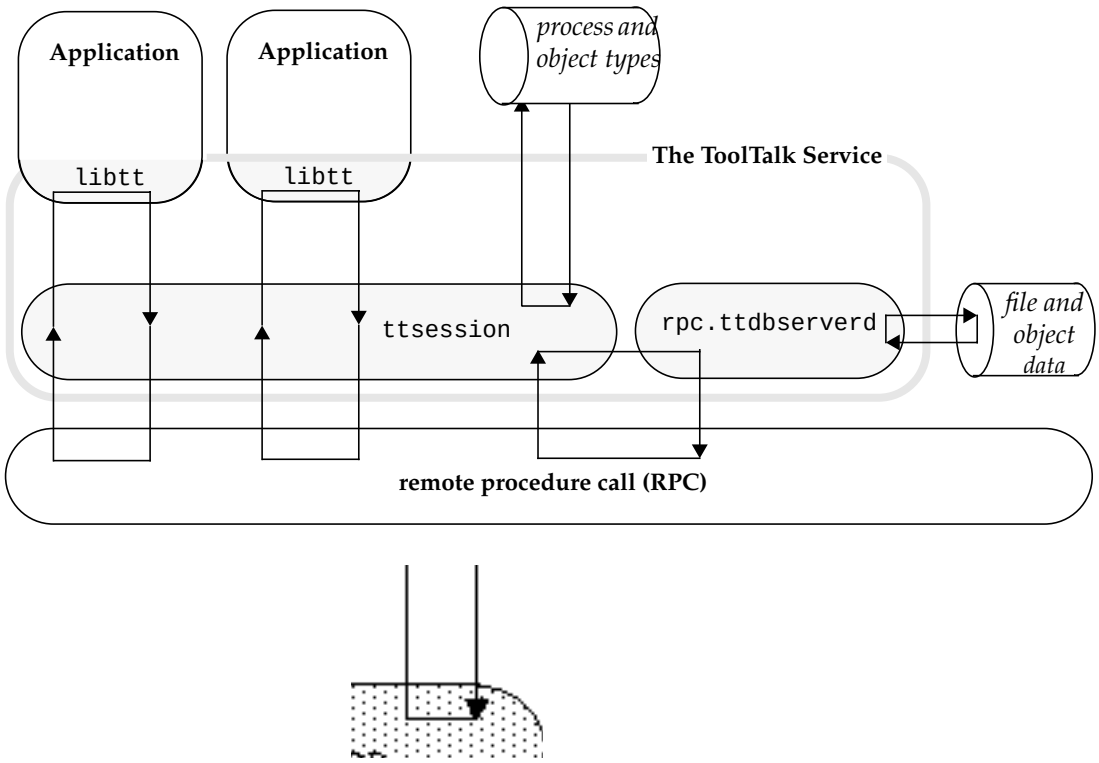


Figure 1-2 ToolTalk Service Architecture

How Applications Use ToolTalk Messages

An important ToolTalk feature is that *sending applications* need to know little about the *receiving applications.* Applications that want to receive messages register their interest in specific types of messages with the ToolTalk service.

To communicate, applications create, send, and receive ToolTalk messages. Sending applications create, fill in, and send a message; the ToolTalk service determines the recipient and delivers the message to the receiving application. Receiving applications retrieve messages, examine the information in the message, and then either discard the message or perform an operation and reply with the results.

Message Patterns

Sending applications need to know little about the receiving application because applications that want to receive messages explicitly state how these messages should appear. This information is registered with ToolTalk in the form of *message patterns*. These message patterns usually match the message protocols that applications have agreed to use. Applications can add more patterns for individual use.

Message patterns are created similar to the way messages are created; the same type of information is used in both. For each type of message an application wants to receive, it obtains an empty message pattern, fills in the attributes, and registers the pattern with the ToolTalk service. Applications provide message patterns to the ToolTalk service at installation time or while the application is running.

When the ToolTalk service receives a message from a sending application, it compares the information in the message to the registered patterns. Once matches have been found, the ToolTalk service delivers copies of the message to all recipients with patterns that match the message.

ToolTalk Messaging Methods

The ToolTalk service provides two methods of addressing messages: process-oriented messages and object-oriented messages.

Process-oriented messages are addressed to processes. Applications that create a process-oriented message address the message to either a specific process or to a particular type of process. Process-oriented messages are a good way for existing applications to begin communication with other applications. Modifications to support process-oriented messages are straightforward and usually take a short time to implement.

Object-oriented messages are addressed to objects managed by applications. Applications that create an object-oriented message address the message to either a specific object or to a particular type of object. Object-oriented messages are particularly useful for applications that currently use objects or that are to be designed around objects. If an existing application is not object-oriented, the ToolTalk service allows applications to identify portions of application data as objects so that applications can begin to communicate about these objects.

Process-Oriented Messages

Process-oriented messages provide a communication path between applications to deliver information to, or to request that an operation be performed by, the process receiving the message. Concepts used in process-oriented messages are described in the following subsections.

Processes

One execution of an application, tool, or program that uses the ToolTalk service is called a *process* in this manual. A process must have initialized and registered with the ToolTalk service.

Process Type Files

Although no process may be running an application, the application can be considered as a potential receiving application by ToolTalk if message patterns and instructions for starting the application are provided in a *process type* (*ptype*) file. The instructions tell the ToolTalk service to perform one of the following operations:

- Start the application and deliver the message
- Queue the message until the application is running
- Discard the message

To make the information available to the ToolTalk service, the ptype file is compiled with the ToolTalk type compiler `tt_type_comp` at application installation time. After the ptype file is compiled, the application instructs the ToolTalk service to retrieve the ptype information as part of the application installation script. If the application does not instruct the ToolTalk service to retrieve the ptype information (either by omission or because of an error condition), you can force the ToolTalk service to read the ptype information. See Chapter 3, "Maintaining Application Information," for more information.

Sessions

A group of processes running in the same X server session or process tree session is called a *session* in this manual. A session also contains an instance of the ToolTalk communication program `ttsession`.

The concept of a session is important in the delivery of messages. Sending applications can "scope" (that is, limit the delivery of) a message to a session and the ToolTalk service will only deliver the message to processes with matching message patterns that refer to the current session. To update message patterns with the current *session identifier (sessid*), applications join the session.

Files

A container for data that is of interest to applications is called a *file* in this manual.

The concept of a file is important in the delivery of messages. Sending applications can scope a message to a file and the ToolTalk service will deliver the message to all processes with matching message patterns that refer to the file. To update message patterns with the current file path name, applications join the file.

The ToolTalk service stores location information about files in a ToolTalk database that is in the same disk partition as the file. This information needs to be up-to-date for the ToolTalk service to deliver messages. You may have to install the ToolTalk database program `rpc.ttdbserverd` on machines that contain files referenced in ToolTalk messages.

Object-Oriented Messages

Object-oriented messages are addressed to objects managed by applications. If your installed applications use this method of messaging, you need to be familiar with process-oriented message concepts plus the ToolTalk concept of objects.

Object Data

Object data is stored in two parts as shown in Figure 1-3.

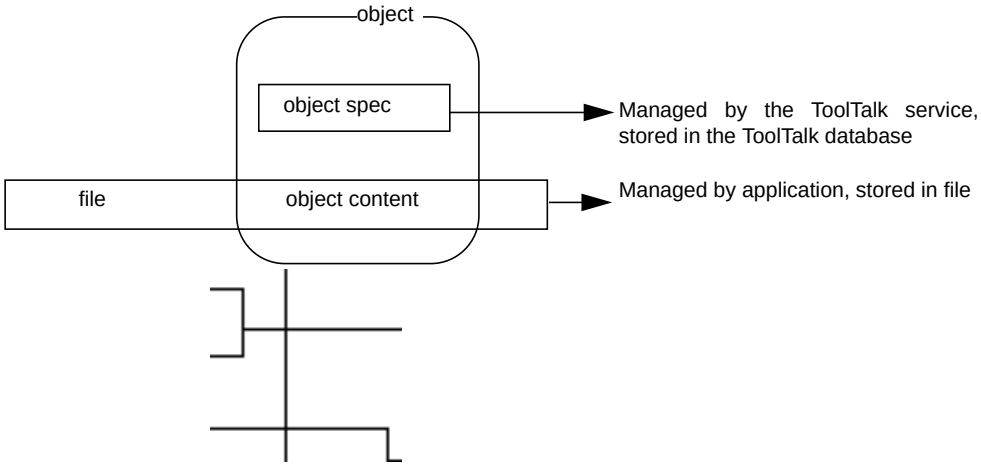


Figure 1-3 ToolTalk Object Data

One part is called the *object content*. The object content is managed by the application that creates or manages the object and is typically a piece, or pieces, of an ordinary file: a paragraph, a source code function, or a range of spreadsheet cells, for example.

The second part is called the *object specification* (*spec*). A spec contains standard properties such as the type of object, the name of the file in which the object contents are located, and the object owner. Applications can also add their own properties to a spec, for example, the location of the object content within a file. Applications create and write specs to the ToolTalk database managed by `rpc.ttdbserver`*d*.

A *ToolTalk object* is a portion of application data for which a ToolTalk spec has been created.

Object Types

When a message is addressed to a specific object or a type of object, the ToolTalk service must be able to determine to which application the message is to be delivered. Applications provide this information in an *object type* (*otype*) file. An otype file includes the ptype name of the application that manages the object and message patterns that pertain to the object.

These message patterns also contain instructions that tell the ToolTalk service what to do if a message is available but the application is not running. In this case, ToolTalk performs one of the following instructions:

- Start the application and deliver the message
- Queue the message until the application is running
- Discard the message

To make the information available to the ToolTalk service, the otype file is compiled with the ToolTalk type compiler `tt_type_comp` at application installation time. As with ptype files, if the application does not instruct the ToolTalk service to retrieve the otype information you can force the ToolTalk service to read otype information. See Chapter 3, “Maintaining Application Information,” for more information.

Object Files

When object-oriented messages are used, a ToolTalk file includes the statement

```
can also contain ToolTalk objects
```

As with files referenced in process-oriented messages, the ToolTalk service stores location information about these object-containing files in the ToolTalk database. It also includes information about the objects contained in a file.

Maintaining ToolTalk Files and Databases

The ToolTalk package contains a set of shell commands you can use to copy, move, and remove ToolTalk files (that is, files mentioned in messages and files that contain ToolTalk objects). After a standard shell command (such as *cp*, *mv*, or *rm*) is performed, the ToolTalk service is notified that a file location has changed.

The ToolTalk package also contains a database check and repair utility for the ToolTalk database, *ttdbck*, that you can use to check and repair your ToolTalk databases.

Chapter 2

Setting Up and Maintaining the ToolTalk Processes

This manual assumes that you have IRIX 5.2 or later successfully installed on your system.

Location of the ToolTalk Service Files

The directories containing ToolTalk service files are shown in Table 2-1.

Table 2-1 Location of ToolTalk Service Files

Directory and File(s)	Description
/usr/ToolTalk/bin/ ttsession	*/usr/ToolTalk/bin/ttsession* is linked from */usr/sbin/ttsession*. *ttsession* is the ToolTalk communication process.
/usr/ToolTalk/bin/ ttcopy ttmv ttrm ttrmdir tttar	The */usr/ToolTalk/bin/tt** files are linked from the corresponding */usr/sbin/tt** symbolic links. These files are shell commands that have been enhanced to inform the ToolTalk service when files that contain ToolTalk objects or files that are the subject of ToolTalk messages are copied, moved, or removed.
/usr/ToolTalk/etc rpc.ttdbserverd	*/usr/ToolTalk/etc/ rpc.ttdbserverd* is linked from *usr/etc/ rpc.ttdbserverd*. *rpc.tdbserverd* stores and manages ToolTalk object specifications and information on files referenced in ToolTalk messages.
/usr/ToolTalk/bin/ ttdbck	*/usr/ToolTalk/bin/ttdbck* is linked fro*m /usr/sbin/ttdbck. ttdbck* is a database check and recovery tool for the ToolTalk databases.
/usr/ToolTalk/bin/ tt_type_comp	*/usr/ToolTalk/bin/tt_type_comp* is linked from */usr/sbin/tt_type_comp*. *tt_type_comp* is a compiler for process types and object types.

Table 2-1 (continued) Location of ToolTalk Service Files

/usr/ToolTalk/lib/ *libtt.so* *libtt.a* */usr/ToolTalk/include* *tt_c.h*	The */usr/ToolTalk/libtt** files are linked from the */usr/libtt** symbolic links. */usr/ToolTalk/include/tt_c.h* is linked from */usr/include/tt_c.h*. These files are the application programming interface (API) libraries and header file that contain the ToolTalk functions used by applications to send and receive messages.
/usr/ToolTalk/man1/ *tt_type_comp.1* *ttcopy.1* *ttmv.1* *ttrm.1* *ttrmdir.1* *ttsession.1* *tttar.1*	*/usr/ToolTalk/man1/* is linked from */usr/man/u_man/manl/ToolTalk/*.
/usr/ToolTalk/man3/ *ttapi.3*	*/usr/ToolTalk/man3/* is linked from */usr/man/p_man/man3/ToolTalk/*.
/usr/ToolTalk/man8/ *rpc.ttdbserverd.8* *ttdbck.8* *ttdbserverd.8*	*/usr/ToolTalk/man8/* is linked from */usr/man/p_man/man8/ToolTalk/*. These are the man pages for the ToolTalk binary files, type compiler, enhanced shell commands, API, and database check utility.

Version

All ToolTalk executables support a `-v` option that prints the version string.

Software Requirements

To find out what other software is required in order to use each ToolTalk subsystem, see Chapter 2 of the ToolTalk release notes.

Chapter 3

Maintaining Application Information

To receive ToolTalk messages, an application must provide information to the ToolTalk service describing the kinds of messages it can respond to. This information, known as message patterns, is provided dynamically either by applications as they run or through ptype and otype files.

Installing Application Types

Installing application types is not a routine task; system administrators should only install type information when an application error condition exists. Ptype and otype files are run through the ToolTalk type compiler at installation time. `tt_type_comp` merges the information into the Type sDatabase. The application then tells the ToolTalk service to read the type information in the Types Database. The following message indicates that an application error condition exists:

```
Application is not an installed type.
```

To install an application's ptype and otype files, follow these steps:

1. Run tt_type_comp on your type file.

   ```
   % tt_type_comp <your-file>
   ```

   `tt_type_comp` runs *your-file* through `cpp`, compiles the type definitions, and merges the information into the Types Database.

Database	Types Database Used
user	*~/.tt/types.xdr*
system	*/etc/tt/types.xdr*
network	*$OPENWINHOME/tt/types.xdr*

Table 3-1 ToolTalk Types Databases

By default, `tt_type_comp` uses the *user* database. To specify another database, use the `-d` option; for example:

```
% tt_type_comp -d user|system|network <your-file>
```

For more information on `tt_type_comp`, see `tt_type_comp(1)`.

2. Force `ttsession` to reread the Types Database.

   To force `ttsession` to reread the Types Database, see the instructions in "Updating the ToolTalk Service"

Examining ToolTalk Type Information

You can examine all type information, only the ptype information, or only the otype information in a specified Types Database. To specify the database you want to examine, use the `-d` option and supply the name of the *user*, *system*, or *network* to indicate the desired database. If the `-d` option is not used, `tt_type_comp` will use the *user* database by default.

- To examine all the ToolTalk type information in a Types Database, enter the following line:

  ```
  % tt_type_comp -p
  ```

  The type information will be printed out in source format.

- To list all ptypes in a Types database, enter the following line:

  ```
  % tt_type_comp -P
  ```

Removing ToolTalk Type Information

You can remove both ptype and otype information from the Types Database.

- Use `tt_type_comp` to remove type information. Enter the following line:

  ```
  % tt_type_comp -d user|system|network -r type
  ```

  For example, to remove a ptype called *Sun_EditDemo* from the network database of a sample application, enter the line:

  ```
  % tt_type_comp -d network -r Sun_EditDemo
  ```

After you remove type information from the Types Database, force `ttsession` to read the Types Database again to bring the ToolTalk service up-to-date. See "Updating the ToolTalk Service"

Updating the ToolTalk Service

Use `ttsession` to force the ToolTalk service to read the type information in the Types Database.

1. Enter the `ps` command to find the process identifier (*pid*) of the `ttsession` process.

   ```
   % ps -elf | grep ttsession
   ```

2. Enter the `kill` command to send a SIGUSR2 signal to `ttsession`.

   ```
   % kill -USR2 <ttsession pid>
   ```

Process Type Errors

One or both of the following conditions exists if application users report the error:

```
Application is not an installed ptype.
```

- The ToolTalk service has not been instructed by the application to read the type information in the Types Database. See “Updating the ToolTalk Service” for instructions on how to force the ToolTalk service to reread type information from the Types Database.
- The application’s ptypes and otypes have not been compiled and merged into the Types Database. See “Installing Application Types” for instructions on how to compile and merge type information.

Chapter 4

Maintaining Files and Objects Referenced in ToolTalk Messages

ToolTalk messages can reference files of interest or ToolTalk objects. The ToolTalk service maintains information about files and objects, and needs to be informed of changes to these files or objects.

The ToolTalk service provides wrapped shell commands to move, copy, and remove files that inform the ToolTalk service of any changes.

Exporting File Systems

To ensure that applications which use the ToolTalk service have access to a mounted file system, always export the actual file system name, not a symbolic link, to the file system. If a user's machine mounts a file system that is exported with a symbolic link, applications that use the ToolTalk service can neither reference files in that file system in ToolTalk messages, nor can they create ToolTalk objects in files in that file system. For example,

Note: The machine `twinpeaks` has a disk mounted on `/home/twinpeaks`.

- A symbolic link is made on the machine `twinpeaks` from `/export/twinpeaks` to `/home/twinpeaks`.
- `/export/twinpeaks` is added to the `/etc/exports` file on machine `twinpeaks`.
- The file system is exported.

When the machine `northwest` mounts `twinpeaks:/export/twinpeaks` on `/mnt`, any attempt by applications to either send ToolTalk messages that reference files in `/mnt`, or to create objects in files in `/mnt`, will fail. To solve the problem:

1. On machine `northwest`, unmount `twinpeaks:/export/twinpeaks.`
2. On machine `twinpeaks`, rename the entry in the `/etc/exports` file to `/home/twinpeaks.`
3. On machine `northwest`, mount `twinpeaks:/home/twinpeaks`.

ToolTalk-Enhanced Shell Commands

The ToolTalk-enhanced shell commands first invoke the standard shell commands with which they are associated (for example, `ttmv` invokes `mv`) and then update the ToolTalk service with the file changes. It is recommended that application users use the ToolTalk-enhanced shell commands when they are working with files that contain ToolTalk objects.

Command	**Definition**	**Synt**tax
ttcopy	Copies files that contain objects that reside on the same machine.	`ttcopy` *source-file destination-file*
ttmv	Moves files that contain objects and removes old version of file.	`ttmv` *old new*
ttrm	Removes files that contain objects.	`ttrm` *file*
ttrmdir	Removes empty directories that are associated with ToolTalk object specs. This command also allows youto create an object spec for a directory. When an object spec is created, the path name of a file or directory is supplied.	`ttrmdir` *directory*
tttar	Archives and de-archives files that contain ToolTalk objects.	`tttar c|t|x` *pathname1 pathname2*

Table 4-1 ToolTalk-Enhanced Shell Commands

The ToolTalk-enhanced shell commands can be changed to look like the standard shell commands. To change the ToolTalk-enhanced shell commands, users can alias the ToolTalk-enhanced shell commands in their shell startup file so they appear as standard shell commands:

```
# ToolTalk-aware shell commands in .cshrc
alias mv     ttmv
alias cp     ttcopy
alias rm     ttrm
alias rmdir  ttrmdir
alias tar    tttar
```

Displaying, Checking, and Repairing Databases

Use the ToolTalk database utility `ttdbck` to display, check, or repair ToolTalk databases.

Information about files and objects in the ToolTalk databases can become out-dated if the ToolTalk-enhanced shell commands are not used to copy, move and remove them. For example, you can remove a file *old_file* that contains ToolTalk objects from the file system with the standard `rm` command. However, because the standard shell command does not inform the ToolTalk service that *old_file* has been removed, the information about the file and the individual objects remains in the ToolTalk database. To remove the file and object information from the ToolTalk database, use the `ttdbck` utility.

Note: ToolTalk databases are typically accessible only to root; therefore, the `ttdbck` utility is normally run as root. See `ttdbck(8)` in the man pages for complete details on how to run this utility.

Initialization Error Messages

The following ToolTalk error messages can occur either when the ToolTalk service, or an application that uses the ToolTalk service, is attempting to start up.

`/bin/sh: application_name: not found`

The start string as installed in the types database does not correspond to an executable file in `$PATH`.

To start the application:

First, ask your user to start the application as they would start it without the ToolTalk service.

After the application has started, ask your user to retry the operation that should have started the application with the ToolTalk service.

`Cannot open display`

ttsession could not contact the server named with the -d *display* option or the $DISPLAY variable.

To open the display:

Verify that the named display is running. (See the `X(1) man page`.)

Verify that the host on which you are running `ttsession` has permission to connect to it. (See the `X(1)`, `xhost(1)`, and `xauth(1) man` pages.

Appendix B

ToolTalk Error Messages

The ToolTalk error and warning message identifiers are allocated as follows.

MESSAGE or GROUP	RANGE
TT_OK	0
TT_WRN_*	1- 511
APP_WRN_*	512 - 1023
TT_WRN_LAST	1024
TT_ERR_*	1025 - 1535
APP_ERR_*	1536 - 2046
TT_ERR_LAST	2047

TT_OK

Message ID	TTERR-0
Catalog String	TT_OK Request successful.
Meaning	Your call was completed successfully.

TT_WRN_NOTFOUND

Message ID	TTERR-1
Catalog String	TT_WRN_NOTFOUND The object was not removed because it was not found.
Meaning	When the ToolTalk service could not find the specified object in the ToolTalk database. The destroy operation did not succeed.

TT_WRN_STALE_OID

Message ID	TTERR-2
Catalog String	TT_WRN_STALE_OID The object attribute in the message has been replaced with a newer one. Update the place from which the object id was obtained.
Meaning	When the ToolTalk service looked up the specified object in the ToolTalk database, it found a forwarding pointer to the object.
Remedy	The ToolTalk service automatically puts the new objid in the message. Use `tt_message_object()` to retrieve the new objid. Update any internal application references to the new objid.

TT_WRN_STOPPED

Message ID	TTERR-3
Catalog String	TT_WRN_STOPPED The query was halted by the filter procedure.
Meaning	The query operation being performed was halted by Tt_filter_function.

TT_WRN_SAME_OBJID

Message ID	TTERR-4
Catalog String	TT_WRN_SAME_OBJID The moved object retains the same objid.
Meaning	The object you moved stayed within the same file system. The ToolTalk service will retain the same objid and update the location.

TT_WRN_START_MESSAGE

Message ID	TTERR-5
Catalog String	TT_WRN_START_MESSAGE This message caused this process to be started. This message should be replied to even if it is a notice.
Meaning	When the ToolTalk service starts your application to deliver a message, you must reply, even if the message is a notice.
Remedy	Use `tt_message_reply()` to reply to the message you received after the process was started by the ToolTalk service.

TT_WRN_APPFIRST

Message ID	TTERR-512
Catalog String	TT_WRN_APPFIRST This code should be unused.
Meaning	This message id marks the beginning of the messages allocated for ToolTalk application warnings.

TT_WRN_LAST

Message ID	TTERR-1024
Catalog String	TT_WRN_LAST This code should be unused.
Meaning	This message id marks the last of the messages allocated for ToolTalk warnings.

TT_ERR_CLASS

Message ID	TTERR-1025
Catalog String	TT_ERR_CLASS The Tt_class value passed is invalid.
Meaning	The ToolTalk service does not recognize the class value you specified.
Remedy	The `Tt_class` values are `TT_NOTICE` and `TT_REQUEST`. Retry the call with one of these values.

TT_ERR_DBAVAIL

Message ID	TTERR-1026
Catalog String	TT_ERR_DBAVAIL A required database is not available. The condition may be temporary, trying again later may work.
Meaning	The ToolTalk service could not access the ToolTalk database needed for this operation.
Remedy	Try the operation again later. Check if the file server or workstation that contains the database is available.

TT_ERR_DBEXIST

Message ID	TTERR-1027
Catalog String	TT_ERR_DBEXIST A required database does not exist. The database must be created before this action will work.
Meaning	The ToolTalk service did not find the specified ToolTalk database in the expected place.
Remedy	Install the `rpc.ttdbserverd` program on the machine that stores the file or object involved in this operation.

TT_ERR_FILE

Message ID	TTERR-1028
Catalog String	TT_ERR_FILE File object could not be found.
Meaning	The file specified does not exist or is not accessible.
Remedy	Check your file path name and retry the operation. Check if the machine where the file is stored is accessible.

TT_ERR_MODE

Message ID	TTERR-1031
Catalog String	TT_ERR_MODE The Tt_mode value is not valid.
Meaning	The ToolTalk service does not recognize the specified mode value.
Remedy	The `Tt_mode` values are `TT_IN`, `TT_OUT`, and `TT_INOUT`. Retry the call with one of these values.

TT_ERR_ACCESS

Message ID	TTERR-1032
Catalog String	TT_ERR_ACCESS An attempt was made to access a ToolTalk object in a way forbidden by the protection system.
Meaning	The user does not have the necessary access to the object and the process, and therefore, cannot perform the operation. For example, the user may not have permission to destroy an object spec.
Remedy	The user needs to gain proper access to the object before the application can perform the operation.

TT_ERR_NOMP

Message ID	TTERR-1033
Catalog String	TT_ERR_NOMP No `ttsession` process is running, probably because `tt_open()` has not been called yet. If this code is returned from `tt_open()` it means `ttsession` could not be started, which generally means ToolTalk is not installed on this system.
Meaning	The `ttsession` process is not available. The ToolTalk service tries to restart `ttsession` if it is not running; this error indicates that the ToolTalk service is either not installed or not installed correctly.
Remedy	Verify that `ttsession` is installed on the machine in use.

TT_ERR_NOTHANDLER

Message ID	TTERR-1034
Catalog String	TT_ERR_NOTHANDLER Only the handler of the message can do this.
Meaning	Only the handler of a message can perform this operation. Your application is not the handler for this message.

TT_ERR_NUM

Message ID	TTERR-1035
Catalog String	TT_ERR_NUM The integer value passed is not valid.
Meaning	An invalid integer value that was very out-of-range was passed to the ToolTalk service. Simple out-of-range conditions, such as requesting the third value of a property that has only two values, return a null value.
Remedy	Check the integer you specified.

TT_ERR_OBJID

Message ID	TTERR-1036
Catalog String	TT_ERR_OBJID The object id passed does not refer to any existing object spec.
Meaning	The ToolTalk service found the objid in the ToolTalk database but it does not reference an existing object.
Remedy	Clean up the ToolTalk database with the ttdbck utility.

TT_ERR_OP

Message ID	TTERR-1037
Catalog String	TT_ERR_OP The operation name passed is not syntactically valid.
Meaning	The specified operation name is null or contains non-alphanumeric characters.
Remedy	Remove any non-alphanumeric characters and retry the operation.

TT_ERR_OTYPE

Message ID	TTERR-1038
Catalog String	TT_ERR_OTYPE The object type passed is not the name of an installed object type.
Meaning	The ToolTalk service could not locate the specified otype.
Remedy	Check the type of the object with `tt_spec_type()`. If the application was recently installed and the ToolTalk service has not re-read the types database, locate the process id for the ttsession, and force the re-read with the USR2 signal. **`% ps -elf | grep ttsession`** **`% kill -USR2 <ttsession pid>`**

TT_ERR_ADDRESS

Message ID	TTERR-1039
Catalog String	TT_ERR_ADDRESS The Tt_address value passed is not valid.
Meaning	The ToolTalk service does not recognize the address value you specified.
Remedy	The `Tt_address` values are `TT_PROCEDURE`, `TT_OBJECT`, `TT_HANDLER`, and `TT_OTYPE`. Retry the call with one of these values.

TT_ERR_PATH

Message ID	TTERR-1040
Catalog String	TT_ERR_PATH One of the directories in the file path passed does not exist or cannot be read.
Meaning	The ToolTalk service was not able to read a directory in the specified file path name.
Remedy	Check the pathname to ensure that the current user has access to the specified directories. Check the machine where the file resides to make sure it is accessible.

TT_ERR_POINTER

Message ID	TTERR-1041
Catalog String	TT_ERR_POINTER The opaque pointer (handle) passed does not indicate an object of the proper type.
Meaning	The pointer you passed does not point at an object of the correct type for this operation. For example, the pointer may point to an integer when a character string is needed.
Remedy	Check the arguments for the ToolTalk function to find what arguments the function expects. Retry the operation with a pointer for a valid object.

TT_ERR_PROCID

Message ID	TTERR-1042
Catalog String	TT_ERR_PROCID The process id passed is not valid.
Meaning	The process identifier you specified is out of date or invalid.
Remedy	Retrieve the default procid with tt_default_procid().

TT_ERR_PROPLEN

Message ID	TTERR-1043
Catalog String	TT_ERR_PROPLEN The property value passed is too long.
Meaning	The ToolTalk service accepts property values of up to 64 characters.
Remedy	Shorten the property value to less than 64 characters.

TT_ERR_PROPNAME

Message ID	TTERR-1044
Catalog String	TT_ERR_PROPNAME The property name passed is syntactically invalid.
Meaning	The property name is too long, contains non-alphanumeric characters, or is null.
Remedy	Check the property name, modify if necessary, and retry the operation.

TT_ERR_PTYPE

Message ID	TTERR-1045
Catalog String	TT_ERR_PTYPE The process type passed is not the name of an installed process type.
Meaning	The ToolTalk service could not locate the specified ptype.

Remedy	If the application was recently installed and the ToolTalk service has not re-read the types database, locate the process id for the ttsession, and force the re-read with the USR2 signal.
	`% ps -elf | grep ttsession`
	`% kill -USR2 <ttsession pid>`

TT_ERR_DISPOSITION

Message ID	TTERR-1046
Catalog String	TT_ERR_DISPOSITION The `Tt_disposition` value passed is not valid.
Meaning	The disposition you passed is not recognized by the ToolTalk service.
Remedy	The `Tt_disposition` values are `TT_DISCARD`, `TT_QUEUE`, and `TT_START`. Retry the call with one of these values.

TT_ERR_SCOPE

Message ID	TTERR-1047
Catalog String	TT_ERR_SCOPE The `Tt_scope` value passed is not valid.
Meaning	The scope you passed is not recognized by the ToolTalk service.
Remedy	The `Tt_scope` values are `TT_SESSION` and `TT_FILE`. Retry the call with one of these values.

TT_ERR_SESSION

Message ID	TTERR-1048
Catalog String	TT_ERR_SESSION The session id passed is not the name of an active session.
Meaning	You specified an out of date or invalid ToolTalk session.
Remedy	Use `tt_default_session()` to obtain the sessid of the current default session or use `tt_initial_session()` to obtain the sessid of the initial session your application was started in.

TT_ERR_VTYPE

Message ID	TTERR-1049
Catalog String	TT_ERR_VTYPE The value type name passed is not valid.
Meaning	The specified property exists in the ToolTalk database but the type of value does not match the specified type; or the value type is not one that the ToolTalk service recognizes. The ToolTalk service supports types of `int` and `string`.
Remedy	Change the type of the value to either `int` or `string` and retry the operation.

TT_ERR_NO_VALUE

Message ID	TTERR-1050
Catalog String	TT_ERR_NO_VALUE No property value with the given name and number exists.
Meaning	The ToolTalk service could not locate the specified property value you specified in the ToolTalk database.
Remedy	Retrieve the current list of properties to find the property you want.

TT_ERR_INTERNAL

Message ID	TTERR-1051
Catalog String	TT_ERR_INTERNAL Internal error (bug)
Meaning	The ToolTalk service has suffered an internal error.
Remedy	Restart all applications that are using the ToolTalk service. Report the error to SGI Customer Support.

TT_ERR_READONLY

Message ID	TTERR-1052
Catalog String	TT_ERR_READONLY The attribute cannot be changed.
Meaning	The attribute your application is trying to change is not owned or writable by the current user.

TT_ERR_NO_MATCH

Message ID	TTERR-1053
Catalog String	TT_ERR_NO_MATCH No handler could be found for this message, and the disposition was not queue or start.
Meaning	The message your application sent could not be delivered. No applications that are running have registered interest in this type of message.
Remedy	Use `tt_disposition_set()` to change the disposition to `TT_QUEUE` or `TT_START` and resend the message. If no recipients are found, no application has registered interest in this type of message.

TT_ERR_UNIMP

Message ID	TTERR-1054
Catalog String	TT_ERR_UNIMP Function not implemented.
Meaning	The ToolTalk function called is not implemented.

TT_ERR_OVERFLOW

Message ID	TTERR-1055
Catalog String	TT_ERR_OVERFLOW Too many active messages (try again later).
Meaning	The ToolTalk service has received the maximum amount of active messages (2000) it can handle properly.
Remedy	Retrieve any messages that the ToolTalk service may be queueing for your application. Send your message again later. `ttsession` can also be started with the `-A` option; specify the maximum number of messages in progress before a TT_ERR_OVERFLOW condition is returned. The default is 2000 messages.

TT_ERR_PTYPE_START

Message ID	TTERR-1056
Catalog String	TT_ERR_PTYPE_START Attempt to launch instance of ptype failed.
Meaning	The ToolTalk service could not start the type of process specified.
Remedy	Check to see that the application that the ptype represents is properly installed.

TT_ERR_APPFIRST

Message ID	TTERR-1536
Catalog String	TT_ERR_APPFIRST This code should be unused.
Meaning	This message id marks the beginning of the messages allocated for ToolTalk application errors.

TT_ERR_LAST

Message ID	TTERR-2047
Catalog String	TT_ERR_LAST This code should be unused.
Meaning	This message id marks the last of the messages allocated for ToolTalk errors.

TT_STATUS_LAST

Message ID	TTERR-2048
Catalog String	TT_STATUS_LAST This code should be unused.
Meaning	This message id marks the last of the messages allocated for ToolTalk status.

Index

R

S

T

U

V

X