The ToolTalk Desktop Services Message Set

F

This appendix contains a description of each of the generic messages that comprise the ToolTalk Desktop Services Message Set. The ToolTalk Desktop Services Message Set is an *open* specification with no royalty or license fees. It will be continuously revised and modified to meet the expanding needs of desktop services developers and users. Send questions, comments, and requests for information to the Desktop Services Messaging Alliance at *ToolTalk_desktop_services@sun.com*.

F.1 General Description of the ToolTalk Desktop Services Message Set

The ToolTalk Desktop Services Message Set conventions apply to any tools in a POSIX or X11 environment. In addition to standard messages for these environments, the Desktop conventions define data types and error codes that apply to all of the ToolTalk inter-client conventions.

F.2 Desktop Definitions and Conventions

This section defines terms and error messages unique to the Desktop Services message set. Specific to the desktop services messages are values associated with fields:

boolean
A vtype for logical values. The underlying data type of boolean is integer; manipulate arguments of this vtype with `tt_*_arg_ival[_set]()` and `tt_*_iarg_add()`. A zero value means *false*; a non-zero value means *true*.

buffer
A volatile, non-shared (for example, in-memory) representation of persistent data.

bufferID
A vtype that uniquely identifies buffers. The underlying data type of bufferID is *string*. To guarantee bufferID uniqueness, use the form

```
<internal_counter> <procID>.
```

messageID
A vtype that uniquely identifies messages. The underlying data type of messageID is *string*; manipulate arguments of this vtype with `tt_*_arg_val[_set]()` and `tt_*_arg_add()`. To guarantee messageID uniqueness, use the form

```
<internal_counter> <procID>
```

`tt_message_id()` returns an opaque string of similar uniqueness. Use `tt_message_id()` to generate a message's messageID; however, the inter-client conventions explicitly include the messageID as a message argument to support inter-operation with other versions of the ToolTalk service.

type
Any vtype that is the name of the kind of objects in a particular persistent-object system. For example, the vtype for the kind of objects in filesystems is `File`; the vtype for ToolTalk objects is `ToolTalk_Object`.

vendor
toolName
toolVersion
Names of arguments. These strings appear in several of the Desktop Service messages. These strings are not defined rigorously; they are intended to present to the user descriptions of these three attributes of the relevant procID.

view
A screen display, such as a (portion of a) window, that presents to the user part or all of a document.

viewID
A vtype that uniquely identifies views. The underlying data type of viewID is *string*. To guarantee viewID uniqueness, use the form

```
<internal_counter> <procID>
```

Errors

Table F-1 describes the Desktop Services error messages; the error messages are listed in order of their message id.

Table F-1 Desktop Services Error Messages

Message ID	Error Message	Error Message String	Description
1538	TT_DESKTOP_ENOENT	No such file or directory	
1549	TT_DESKTOP_EACCES	Permission Denied	
1558	TT_DESKTOP_EINVAL	Invalid argument	An argument's value was not valid; for example, a locale in `Set_Locale` that is not valid on the handler's host. Use this error status only when a more-specific error status does not apply.
1571	TT_DESKTOP_ENOMSG	No message of desired type	A messageID does not refer to any message currently known by the handler.
1610	TT_DESKTOP_EPROTO	Protocol error	A message was not understood because: a. A required argument was omitted. b. An argument had the wrong vtype, or the vtype is not allowed in this message; for example, the vtype `boolean` in the `Get_Geometry` message. c. An argument's value was not legal for its vtype; for example, negative values for width in the `Set_Geometry` message. d. An argument's value was not legal for this message; for example, the `PATH=/foo` variable in `Get_Environment` message. In general, this error status indicates that the message is malformed.

Table F-1 Desktop Services Error Messages (Continued)

Message ID	Error Message	Error Message String	Description
1688	TT_DESKTOP_CANCELED	Operation was canceled	The operation was canceled because of direct or indirect user intervention. An example of indirect intervention is termination of the handling process caused by the user, or receipt of a Quit() request. (All messages should be taken as authentically representing the wishes of the user whose uid is indicated by tt_message_uid().)
1689	TT_DESKTOP_ENOTSUP	Operation not supported	The requested operation is not supported by this handler. This error indicates that a handler assumes that, if it rejects a request, no other handler will be able to perform the operation. For example, a request such as Set_Iconified() or a request that refers to a state (such as a bufferID) that is managed by this handler alone. A request failed with this error distinguishes the case of an incompletely-implemented handler from the case of the absence of a handler. **Note:** Do not use TT_ERR_UNIMP in place of TT_DESKTOP_ENOTSUP as TT_ERR_UNIMP means that a particular feature of ToolTalk itself is not implemented.
1699	TT_DESKTOP_UNMODIFIED	Operation does not apply to unmodified entities	

Warnings

The vtype namespace for persistent objects currently only contains `File` and `ToolTalk_Object`. Vendors who want to define a type should either give it a vendor-specific name or register it through SunSoft's Developer Integration Format Registration program. SunSoft can be reached at 1-800-227-9227.

F.3 The ToolTalk Desktop Services Message Set

This section contains a description of each of the generic messages which constitute the ToolTalk Desktop Services Message Set.

Created, Deleted (Notice)

Notification that entities (for example, files) have been created or deleted.

Synopsis

```
[file] Created(in type ID[...]);

[file] Deleted(in type ID[...]);
```

Description

The Created notice is sent whenever a tool creates or deletes one or more entities that may be of interest to other tools.

Required Arguments

type ID
: The identity of the created entity. If more than one entity are created in the same logical event, extra ID arguments may be present.

  When *type* is File, each non-empty ID argument is the name of an entry which has been created in the directory named in the message's file attribute. (Each argument is, therefore, a single, final component of a pathname.)

  When *type* is File and this argument is empty (that is, has a value of (char *)0), it refers to the file or directory named in the message's file attribute.

Optional Arguments

type ID
: Extra instances of this argument may be included.

Do_Command (Request)

Requests in a tool's native command language that a command be performed.

Synopsis

```
Do_Command(  in      string        command,
             out     string        results
             [in     messageID     counterfoil] );
```

Description

The Do_Command message requests that the receiving tool perform a command. The request is stated in the receiving tool's native command language.

When the request includes the optional *counterfoil* argument, the handler can send an immediate point-to-point status notice back to the requesting tool if the requested operation is expected to require an extended amount of time.

Required Arguments

string command
: The command being requested to be performed.

string results
: The results of the completed command. The results are returned as if the command had been executed locally to the requesting tool.

Optional Arguments

messageID counterfoil

Unique string created by the message sender (typically by concatenating a counter and a procID) to give both sender and receiver a way to refer to this request in other correspondence. Include this argument if the sender anticipates a need to communicate with the handler about this request before it is completed; for example, to cancel it.

When this argument is included and the handler determines that an immediate reply is not possible, then the handler should immediately send at least one Status notice point-to-point back to the requestor to identify itself to the requestor.

Warnings

This request allows tools to provide a message interface to functionality that is not supported through any standard (or even tool-specific) message interface. This message, therefore, constitutes a deprecated interface when the intended function is available through an existing message interface.

F

Get_Modified (Request)

Asks whether an entity (for example, a file) has been modified.

Synopsis

```
[file]  Get_Modified(intype          ID,
                out     boolean      modified);
```

Description

The Get_Modified message asks whether any tool has modified a volatile, non-shared for example, in-memory) representation of the persistent state of an entity (such as a file) with the intention of eventually making that representation persistent. Therefore, a tool should register a dynamic pattern for this request when it has modified an entity of possible shared interest.

Required Arguments

type ID
: The identity of the entity that may have been modified.

  When *type* is `File`, this argument is empty (that is, it has a value of (char *) 0) and references the file or directory named in the message's file attribute.

boolean modified
: The boolean value that indicates whether a volatile, non-shared (for example, in-memory) representation of the entity has been modified with the intention of eventually making that representation persistent.

Errors

TT_ERR_NO_MATCH
: The Get_Modified request failed because no handler was found and the named entity is assumed not to be modified.

Get_Status (Request)

Requests that a tool's current status be returned.

Synopsis

```
Get_Status(  out     string         status,
             out     string         vendor,
             out     string         toolName,
             out     string         toolVersion
             [in     messageID      operation2Query]);
```

Description

The Get_Status message retrieves either the current status of a tool or the current status of a specific operation that is being performed by a tool.

Required Arguments

string status
The status to be retrieved.

string vendor
The name of the vendor of the receiving tool.

string toolName
The name of the receiving tool.

string toolVersion
The version of the receiving tool.

Optional Arguments

messageID operation2Query
The ID of the request that initiated the operation the status of which is being requested.

F

Get_Sysinfo (Request)

Retrieves information about a tool's host.

Synopsis

```
Get_Sysinfo( out     string        sysname,
             out     string        nodename,
             out     string        release,
             out     string        version,
             out     string        machine,
             out     string        architecture,
             out     string        provider,
             out     string        serial);
```

Description

The Get_SysInfo message retrieves information about the receiver's host.

Required Arguments

string sysname
The name of the host's operating system.

string nodename
The name of the host.

string release
string version
Vendor-determined information about the host's operating system.

string machine
A vendor-determined name that identifies the hardware on which the operating system is running (such as sun4, sun4c, or sun4m).

string architecture
A vendor-determined name that identifies the instruction set architecture of the host (such as sparc, mc68030, m32100, or i80486).

string provider
The name of the hardware manufacturer.

string serial
The ASCII representation of the hardware-specific serial number of the host.

See Also

sysinfo(2), umane(2)

Modified, Reverted (Notice)

Notification that an entity (for example, a file) has been either modified or reverted to its prior state.

Synopsis

```
[file] Modified(in   type          ID);

[file] Reverted(in   type          ID);
```

Description

The Modified message notifies interested tools whenever a tool first makes changes to a volatile, non-shared (for example, in-memory) representation of the persistent state of an entity (such as a file). The Reverted message notifies interested tools whenever a tool discards the modifications made to a volatile, non-shared (for example, in-memory) representation of the persistent state of an entity (such as a file).

Required Arguments

type ID
The identity of the modified or reverted entity.

When *type* is File, this argument is empty (that is, has a value of (char *)0) and refers to the file or directory named in the message's file attribute.

Moved (Notice)

Notification that an entity (for example, a file) has been moved.

Synopsis

```
[file] Moved(in      type          oldID,
             in      type          newID);
```

Description

The Moved message notifies interested tools whenever a tool changes the location of a persistent entity.

Required Arguments

type newID
: The new identity of the moved entity.

  When *type* is File, this argument is empty (that is, has a value of (char *)0), and refers to the file or directory named in the message's file attribute.

type oldID
: The old identity of the moved entity.

  When *type* is File, this argument is either an absolute pathname, or a pathname relative to the directory named in (or containing) the path in the message's file attribute.

Pause, Resume (Request)

Requests the specified tool, operation, or data performance to pause or resume.

Synopsis

```
Pause(      [in     messageID     operation] );

Pause(      in      bufferID      docBuf );

Resume(     [in     messageID     operation] );

Resume(     in      bufferID      docBuf
            [in     locator       whither
            |in     vector        duration] );
```

Description

The Pause or Resume messages requests that the specified tool, operation, or data performance pause or resume, respectively.

- If the optional *operation* argument is included, the handler should pause or resume the operation that was invoked by the specified request. Use a Tt_address of TT_HANDLER to send this form of the request.
- If the optional *docBuf* argument is included, performance of the data in the specified buffer should be paused or resumed. Use a Tt_address of TT_PROCEDURE to send this form of the request.
- If both of the optional arguments are omitted, the handling procid should pause or resume its operations. Use a Tt_address of TT_HANDLER to send this form of the request.

Caution – The Pause and Resume requests may also be sent as a multicast notices; however, the consequences can be severe and unexpected.

Optional Arguments

bufferID docBuf
The buffer in which data performance is to be paused or resume.

messageID operation
The request to be paused.

locator whither
The buffer location to which performance is to be resumed.

vector duration
The duration for which performance is to be resumed.

Note – If neither the *whither* nor the *duration* argument is included in this message, the performance is resumed indefinitely.

Errors

TT_ERR_NOMATCH
The bufferID may not be valid; no editor has a pattern handling this request for docBuf.

TT_DESKTOP_EINVAL
The value for the *whither* is not a legal locator for the media type of the document in docBuf.

TT_DESKTOP_EINVAL
The destination is not a legal vector for the media type of the document in docBuf.

TT_DESKTOP_EFAULT
The value for the *whither* argument is not a valid locator for the document in docBuf.

TT_DESKTOP_EFAULT
The value for the *duration* argument is not a valid vector for the document in docBuf.

TT_DESKTOP_ENOMSG
The operation does not refer to any message currently known by the handler.

Quit (Request)

Requests that an operation, or an entire tool, terminate.

Synopsis

```
Quit(       in      boolean       silent,
            in      boolean       force
            [in     messageID     operation2Quit]);
```

Description

Without the optional *operation2Quit* argument, this request asks the recipient procID to quit. If the request succeeds, one or more ToolTalk procID's should call tt_close(), and zero or more processes should exit. ("Zero or more process" are indicated because a single process can instantiate multiple independent procID's, and a single procID can conceivably be implemented by a set of cooperating processes.)

With the optional *operation2Quit* argument, this request asks the recipient to terminate the indicated request. (Whether the terminated request must be failed depends on its semantics. Often, termination can be considered to indicate that the requested operation has been carried out to the requestor's satisfaction.)

This request should be failed (and the status code set appropriately) when the termination is not performed -- for example, because the *silent* argument was false and the user canceled the quit operation.

Caution – The Quit request may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Required Arguments

boolean silent
: Boolean value that indicates whether the recipient tool is allowed to block on user input before terminating itself, or the indicated operation. If this value is *false*, the handler is not required to seek user input.

boolean force
Boolean value that indicates whether the recipient tool should terminate itself even if circumstances are such that the tool ordinarily would not terminate under them.

For example, a tool might have a policy of not quitting with unsaved changes unless the user has been asked whether the changes should be saved. When this argument is true, such a tool should terminate even when doing so would lose changes that the user has not been asked about saving.

Optional Arguments

messageID operation2Quit
The request that should be terminated. For a request to be terminable, an (optional) counterfoil messageID shall have been included in the request, and the handler shall have sent a Status notice back to the requestor (thus identifying itself to the requestor).

Errors

TT_DESKTOP_ECANCELED
The Quit request was over-ridden by the user.

TT_DESKTOP_ENOMSG
The operation2Quit argument does not refer to any message currently known by the handler.

Raise, Lower (Request)

Raises or lowers a tool's window(s) to the front or back, respectively.

Synopsis

```
Raise(      [in     messageID      commission...]
            [in     viewID         view2Raise...]);

Lower(      [in     messageID      commission...]
            [in     viewID         view2Lower...]);
```

Description

The Raise and Lower message raise or lower, respectively, the window(s) associated with the recipient's procid. If any optional arguments are present, only the indicated window(s) are raise or lowered.

Caution – The Raise and Lower requests may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Optional Arguments

messageID commission
: The identifier of the message (if any) that resulted in the creation of the raised or lowered window(s).

viewID view2Raise
viewID view2Lower
: The identifier of the view whose associated window(s) is (are) be raised or lowered.

Save, Revert (Request)

Saves or discards any modifications to an entity (for example, a file).

Synopsis

```
[file]      Save(   in          typeID);
[file]      Revert( in          typeID);
```

Description

The Save and Revert messages requests that any pending, unsaved modifications to a persistent entity (such as a file) be saved or discarded, respectively.

Required Arguments

type ID
The identity of the entity to save or revert.

When *type* is `File`, this argument is empty (that is, it has a value of (char *) 0) and references the file or directory named in the message's file attribute.

Errors

TT_DESKTOP_UNMODIFIED
The entity had no pending, unsaved modifications.

`TT_DESKTOP_ENOENT`
The file to save or revert does not exist.

Saved(Notice)

Notification that an entity (such as a file) has been saved to persistent storage.

Synopsis

```
[file] Saved(in      type          ID);
```

Description

The Saved message notifies interested tools whenever a tool saves an entity (such as a file) to persistent storage.

Required Arguments

type ID
: The identity of the saved entity.

When *type* is File, this argument is empty (that is, has a value of (char *)0), and refers to the file or directory named in the message's file attribute.

Set_Environment, Get_Environment (Request)

Requests that a tool's environment either be set or retrieved.

Synopsis

```
Set_Environment(     in              stringvariable,
                     in              stringvalue
                     [...]);

Get_Environment(     in              stringvariable,
                     out             stringvalue
                     [...]);
```

Description

The Set_Environment and Get_Environment messages request that the value of the indicated environment variable(s) either be replaced or reported, respectively.

Caution – The Set_Environment request may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Required Arguments

string variable
The name of the environment variable to be set or retrieved.

string value
The value of the environment variable to be set or retrieved.

- If this argument does not contain a value for the Set_Environment request, the variable is removed from the environment. It is not considered an error if the specified variable does not exist.
- If this argument does not contain a value when used in the Get_Environment request, the variable was not present in the receiving tool's environment. This condition is not considered an error.

Optional Arguments

string variable
string value
Extra pairs of these arguments may included.

Set_Geometry, Get Geometry (Request)

Requests that a tool's on-screen geometry either be set or retrieved.

Synopsis

```
Set_Geometry(inout    width          w
             inout    height         h
             inout    xOffset        x
             inout    yOffset        y
             [in      messageID      commission]
             [in      viewID         view2Set]);

Get_Geometry(out      width          w
             out      height         h
             out      xOffset        x
             out      yOffset        y
             [in      messageID      commission]
             [in      viewID         view2Get]);
```

Description

The Set_Geometry and Get_Geometry messages request that the value of the on-screen geometry of the optionally-specified window, or the value of the on-screen geometry of the window primarily associated with the receiving tool's procID if no window is specified, be either set or retrieved (respectively).

Required Arguments

width w
height h
xOffset x
yOffset y

The integer geometry values in pixels.

The return values for the Get_Geometry request are the actual new values, not the requested new values.

Note – Negative offset values are interpreted according to `X11` rules.

Optional Arguments

messageID commission
The identifier of the message (if any) that resulted in the creation of the set or retrieved window(s).

viewID view2Set
viewID view2Get
The identifier of any view associated with the window(s) that is (are) to be set or retrieved.

Set_Iconified, Get_Iconified(Request)

Requests that a tool's iconic state be set or retrieved.

Synopsis

```
Set_Iconified(inout   boolean        conic
              [in     messageID      commission]
              [in     viewID         view2Iconify]);

Get_Iconified(out     boolean        iconic
              [in     messageID      commission]
              [in     viewID         view2Query]);
```

Description

The Set_Iconified and Get_Iconified messages request that the value of the iconic state of the optionally-specified window, or the iconic state of the window primarily associated with the receiving tool's procID if no window is specified, be either set or retrieved (respectively).

Caution – The Set_Iconified and Get_Iconified requests may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Required Arguments

boolean iconic
: The boolean value that indicates whether the specified window is iconified.

Optional Arguments

messageID commission
: The identifier of the message (if any) that resulted in the creation of the iconified or queried window(s).

viewID view2Iconify
viewID view2Query
: The identifier of any view associated with the window(s) that is (are) to be iconified or queried.

Set_Locale, Get_Locale (Request)

Sets or retrieves a tool's locale.

Synopsis

```
Set_Locale(  in      string        category,
             in      string        locale
             [...]);

Get_Locale(  in      string        category,
             out     string        locale
             [...]);
```

Description

The Set_Locale and Get_Locale messages replace or report (respectively) the locale of the POSIX locale categories.

Caution – The Set_Locale request may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Required Arguments

string category
The locale category to set or retrieve.

A locale category is a group of data types whose formatting varies according to locale; for example, ANSI C and X/OPEN locale categories include:

- LC_CTYPE
- LC_NUMERIC
- LC_TIME
- LC_COLLATE
- LC_MONETARY
- LC_ALL
- LC_MESSAGES (Solaris-specific)

string locale
The name of the current locale of the indicated category, or the locale to which to set the indicated category; example of these locales defined in UNIX SVR4 are "C", "de", "fr", and "it".

Optional Arguments

string category
string locale
Extra pairs of these arguments may be included.

Set_Mapped,Get_Mapped(Request)

Requests that a tool's mapping to the screen be set or retrieved.

Synopsis

```
Set_Mapped(   inout    boolean        mapped
              [in      messageID      commission]
              [in      viewID         View2Map]);

Get_Mapped(   out      boolean        mapped
              [in      messageID      commission]
              [in      viewID         view2Query]);
```

Description

The Set_Mapped and Get_Mapped messages request that value of the mapped state of the optionally-specified window, or the mapped state of the window primarily associated with the receiving tool's procID if no window is specified, be either set or retrieved (respectively).

Caution – The Set_Mapped request may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Required Arguments

boolean mapped
: The boolean value that indicates whether the specified window is mapped to the screen.

Optional Arguments

messageID commission
: The identifier of the message (if any) that resulted in the creation of the set or retrieved window(s).

viewID view2Map
viewID view2Query
: The identifier of any view associated with the window(s) that is (are) to be set or retrieved.

Set_Situation, Get_Situation

Requests that a tool's current working directory be set or reported.

Synopsis

```
Set_Situation(in     string        path );

Get_Situation(out    string        path );
```

Description

The Set_Situation and Get_Situation messages request that value of the current working directory be either set or reported (respectively).

Caution – The Set_Situation request may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Required Arguments

string path
: The pathname of the working directory that the recipient is either using or is to use.

Set_XInfo, Get_XInfo (Request)

Requests that a tool's X11 attributes be set or retrieved.

Synopsis

```
Set_XInfo(    inout    string         display,
              inout    string         visual,
              inout    integer        depth
              [in      messageID      commission]
              [inout   string         resourceName,
              inout    string         resourceVal,...]);

Get_XInfo(    out      string         display,
              out      string         visual,
              out      integer        depth
              [in      messageID      commission]
              [in      string         resourceName,
              out      string         resourceVal,...]);
```

Description

The Set_XInfo and Get_XInfo messages request that the X11 attributes of the optionally-specified window, or the X11 attributes of the window primarily associated with the receiving tool's procID if no window is specified, be either set or retrieved (respectively).

Required Arguments

string display
An X11 display.

Note – Since the handler may be running on a different host, use the value `hostname:n[.n]` rather than `:n[.n]`.

---

string visual
An X11 visual class, which determines how a pixel will be displayed as a color. Values include:

StaticGray
GrayScale
StaticColor
PseudoColor
TrueColor
DirectColor

integer depth
The number of bits in a pixel.

Optional Arguments

string resourceName
string resourceVal
An X11 resource name and resource value.

messageID commission
The ID of the message with respect to which X11 attributes are being set or reported. This is useful to the extent that the handler employs different attributes for the different operations it may be carrying out.

Signal(Request)

Requests that a (POSIX-style) signal be sent to a tool.

Synopsis

```
Signal(      in      integer        theSignal);
```

Description

The Signal message requests that the receiving tool's procID send the indicated signal to itself.

Required Arguments

integer theSignal
The signal to be sent.

Caution – The Signal request may also be sent as a multicast notice; however, the consequences can be severe and unexpected.

Started, Stopped (Notice)

Notification that a tool has started or terminated.

Synopsis

```
Started(     in      string      vendor,
             in      string      toolName,
             in      string      toolVersion);

Stopped(     in      string      vendor,
             in      string      toolName,
             in      string      toolVersion);
```

Description

The Started and Stopped messages notify interested tools whenever a tool starts or terminates, respectively.

Required Arguments

string vendor
The name of the vendor of the started or terminated tool.

string toolName
The name of the started or terminated tool.

string toolVersion
The version of the started or terminated tool.

Status(Notice)

Notification that a tool has status information to announce.

Synopsis

```
Status(      in      string        status,
             in      string        vendor,
             in      string        toolName,
             in      string        toolVersion
             [in     messageID     commission]);
```

Description

The Status message notifies interested tools of a tool's general status information.

Required Arguments

string status
The status which is being announced.

string vendor
The name of the vendor of the tool whose status is being announced.

string toolName
The name of the tool whose status is being announced.

string toolVersion
The version of the tool whose status is being announced.

Optional Arguments

messageID commission
The ID of the request, if any, that initiated the operation the status of which is being announced.

F