

TOLIS Tape Tools™

User's Guide



The utilities included in the TOLIS Tape Tools package include:

- tapectl™** - Used to control tape drive operations
- libctl™** - Used to control tape library operations
- taperead™** - Used to read raw data off of a tape
- tapewrite™** - Used to write raw data to a tape
- tapecopy™** - Copy tapes between two tape drives

These tools provide access to tape drives and tape libraries that is not provided by default under Mac OS X.



Table of Contents

Controlling Tape Drives - tapectl.....	5
Example tapectl commands and their results.....	6
tapectl command breakdown.....	7
Controlling a Tape Library - libctl.....	10
Example libctl commands and their results.....	11
libctl command breakdown.....	12
OS X Standard IO Streams - a One Page Primer.....	14
Reading a tape with taperead.....	15
taperead examples with tar.....	16
Important tar notes.....	16
taperead examples with cpio.....	18
Writing a tape with tapewrite.....	19
Duplicate Tapes with tapecopy.....	20
TOLIS Group Support Notes.....	21
TOLIS Tape Tools License Agreement.....	22

Controlling Tape Drives - tapectl

Tape drives control their media in one of two manners - rewinding mode and non-rewinding mode. In rewinding mode, the media is opened, if the media is not at its beginning, it is rewound, the user operation is performed, and the media is rewound to its beginning. In non-rewinding mode, the media is opened wherever it is positioned, the operation is performed, and the media position remains where the last command left it. The first mode is useful only for writing or reading a single data set from the beginning of the media. The second mode allows much greater flexibility in how a tape is used.

All of the TOLIS Tape Tools utilities use the non-rewinding mode of access.

The tapectl utility provides the following tape operations:

display	- displays information on all available tape devices
inquiry	- displays SCSI inquiry details for the selected drive
status	- display current drive status
fsf x	- forward space filemark (x = count; 1 is default)
bsf x	- back space filemark (like fsf, but backwards)
seod	- space to end of data for appends
erase	- short erase a tape
lerase	- long (secure) erase - takes hours!
rewind	- rewinds a tape
rewoff	- rewind the tape and unload it
reten	- retention a tape (QIC and TRAVAN tapes)
tell	- report current logical tape block
seek x	- seek to logical tape block 'x'
setblk x	- set drive block size to x (x is required, 0 = variable block)
alert	- display Tape Alert information (if supported and available)
unload	- same as rewoff

Example tapectl commands and their results

```
# tapectl -f ntape0 display
Available Tape Devices:
  ntape0: IBM ULTRIUM-TD1 25D4
  ntape1: IBM ULTRIUM-TD1 25D4

# tapectl -f ntape1 inquiry
Vendor = IBM      , Model = ULTRIUM-TD1
Revision Level = 25D4

# tapectl -f ntape1 status
Medium Type: Unknown
Density Code: 0x40 - DLT1 40 GB, or LTO 1
BlockSize: 0
At block 55259
```

Most other tapectl commands perform their operations silently with the return code of the command indicating success or failure.

```
# tapectl -f ntape1 rewind
# echo $?
0
```

This return code (or exit code) of zero indicates that the rewind was successful. Any non-zero return code indicates an error occurred. In most instances, an error will also return text details if the -v (verbosity) option is included in the command:

```
# tapectl -f ntape0 -v erase
Erasing media in drive ... Device Not Ready, No tape inserted
# echo $?
2
```

The return code from this command was 2, and the error message indicates that we attempted to perform a tape operation, but no tape was in the drive.

The following commands can be issued when no tape is in the drive with successful results:

```
display, inquiry, status, setblk, and alert
```

All other tapectl commands will return an error if there is not tape in the drive.

tapectl command breakdown

display

`tapectl display`

The display command searches all busses on your system and reports any type 1 SCSI devices that are located. These devices can be SCSI, Fibre-Channel, Firewire, or USB. We use a naming convention of `ntapeX` – where X is a number starting at zero. The default tape device for all command is `ntape0`.

inquiry

`tapectl inquiry`

The inquiry command issues a SCSI INQUIRY command to the defined device and displays the data returned by the tape drive.

status

`tapectl status`

The status command gets information from the tape drive and reports the load status, data block size, current block and any SCSI sense messages.

fsf

`tapectl fsf 4`

The fsf, or Forward Space Filemark command moves the tape forward from its current position to the next filemark. The tape is left positioned on the end of tape (EOT) side of the filemark ready to read or write. You can specify multiple filemarks by providing a number argument after the fsf command.

bsf

`tapectl bsf 3`

The bsf, or Back Space Filemark command moves the tape backwards from its current position to the previous filemark. The tape is left at the EOT side of the filemark. You can specify multiple filemarks by providing a number argument after the bsf command.

seod

`tapectl seod`

The seod, or Space to End Of Data command moves the tape forward to the end of currently recorded data. This allows you to position the tape in preparation for writing additional data.

erase

`tapectl erase`

The erase command uses the SCSI short erase function to clear the contents of the tape. The short erase simply blanks the header on the tape and writes an end of data (EOD) marker at the beginning of the

tape. It does not physically erase any data that was previously written on the tape.

lerase

`tapectl lerase`

The lerase, or Long Erase command actually erase all data on the tape before resetting the EOD marker. While this erase method is more secure than the short erase above, it can take a very long time for a long erase to complete. For example, a 230M SDX3-100 AIT3 tape will take almost 5-1/2 hours to complete the long erase. A DLT VS80 tape will take a little over 2 hours. For most environments, the short erase provides enough security as it takes a specially modified tape drive to read beyond the EOD marker.

rewind

`tapectl rewind`

The rewind command returns the tape to its beginning.

rewoff

`tapectl rewoff`

The rewoff, or Rewind Offline command rewinds the tape and ejects it from the drive.

reten

`tapectl reten`

The reten command is only useful for QIC or TRAVAN tape mechanisms. These tapes sometimes require retentioning to reset the tape travel path within the cartridge shell. If you are having problems reading a known good TRAVAN or QIC tape, running two or three retention passes will usually return the tape to readable status. The reten command does nothing if issued to other types of tape drives such as DAT, AIT, DLT, or LTO.

tell

`tapectl tell`

The tell command reports the current block location on the tape. Tapes write data in blocks. Using tell can make it easier to locate a specific file or backup set on the tape.

seek

`tapectl seek 32452`

The seek command tells the drive to position the tape to the block specified by the numeric argument provided.

setblk

`tapectl setblk 0`

The setblk, or set block command sets the physical block size that data is separated into when it is written to the tape surface. For SCSI and

Fibre-Channel drives, both variable and fixed block sizes are supported. For ATAPI based drives (usually found in Firewire and USB versions of a drive), only fixed block operation is supported. For most SCSI drives, variable block mode is the most efficient operating mode (setblk 0).

alert

tapectl alert

The alert command returns the Tape Alert™ status from a compatible tape drive. The Tape Alert status can provide very specific information concerning the health of your tape drive and your media.

unload

tapectl unload

The unload command is a synonym for rewoff

wfm

tapectl wfm

The wfm command writes a filemark at the current head location on the tape media. By default, when you use the tapewrite tool to write data onto the tape, a filemark is automatically generated when you complete the write process and device is closed.

Controlling a Tape Library - libctl

A tape library allows you to automate the use of multiple tapes and even multiple tape drives without direct user intervention. Tape libraries are often referred to by the names Library, Autoloader, or Changer, but these all refer to the same basic idea – robotic control of loading and unloading of a tape drive within a self contained cabinet. The general distinction is that an Autoloader is usually a smaller unit supporting 10 or fewer tape slots and a single tape drive, while a Library or Changer supports 10s or even 1000s of tape slots and one or more tape drives.

Additionally, it is important to note that many libraries support different access modes. These modes include SCSI – or Random, Console, LCD, and Sequential modes. For proper software operation with libctl, your library should be set to SCSI – or Random mode. For information on the other modes and how you interface with them, please refer to your library's documentation.

The TOLIS Tape Tools libctl utility provides the mechanism to control all of the robotic operations of a library. Commands provided by libctl include:

display	- displays available changer devices
inquiry	- displays the SCSI inquiry info for the selected library
status	- displays info on library queried.
initialize	- (re)initialize elements / read barcodes.
inventory	- Displays load settings for drive(s) and tapes.
load [s] [d]	- Load a tape from slot [s] to drive [d].
unload [d] [s]	- Unload tape from drive [d] to slot [s].
move <fs> <ts>	- Moves tape in slot fs to slot ts. fs and ts are required.
export <fs> [ie]	- Places tape from slot fs into the ie (mailslot) specified.
import [ie] <ts>	- Loads tape in the ie (mailslot) to slot ts. export and import default to the first ie slot.
unlock	- Unlocks the i/e port (mailslot) for addition of tape(s).

Example libctl commands and their results

```
# libctl display
Available Tape Changers:
  changer0: SONY LIB-162 01m6

# libctl status
Vendor = SONY      , Model = LIB-162
Revision Level = 01m6
Unit has barcode reader
Robots: 1 (0), Drives: 1 (82), Tape Slots: 16 (1 - 16),
  No Import/Export slots.
  Drive   0: Empty
    Slot   1: Full : Ready : A01JEN
    Slot   2: Full : Ready : A01JEG
    Slot   3: Full : Ready : A01JED
    Slot   4: Full : Ready : ABD123
    Slot   5: Full : Ready : ABD120
    Slot   6: Full : Ready : ABD125
    Slot   7: Full : Ready : ABD126
    Slot   8: Full : Ready : A01JEE
    Slot   9: Full : Ready : A01JG4
    Slot  10: Empty
    Slot  11: Empty
    Slot  12: Empty
    Slot  13: Empty
    Slot  14: Empty
    Slot  15: Empty
    Slot  16: Full : Ready : No Bar Code

# libctl -v load 3 0
Vendor = SONY      , Model = LIB-162
Revision Level = 01m6
Unit has barcode reader
Robots: 1 (0), Drives: 1 (82), Tape Slots: 16 (1 - 16),
  No Import/Export slots.
Move Element Called 0, 3, 82
Move Element Complete.
```

As with `tapectl`, all `libctl` commands will provide a return code. A zero indicates complete success while non-zero indicates an error has occurred.

libctl command breakdown

display

`libctl display`

The display command lists the libraries available according to your OS X. Libraries are type 8 SCSI devices. To provide a standardized naming convention, we refer to the devices as changerX where X is a number starting at zero. The default device for all commands is changer0.

inquiry

`libctl inquiry`

The inquiry command issues a SCSI INQUIRY command to the defined device and displays the data returned by the tape library.

status

`libctl status`

The status command returns the current status of the selected tape library. This information includes vendor info, number of drives, tape slots, and import/export slots, as well as the status of each of these elements.

initialize

`libctl initialize`

The initialize command performs a full INITIALIZE ELEMENT STATUS operation on the entire library. For smaller libraries, this command will finish very quickly, but for a larger library (100+ slots), it can take many minutes to complete.

inventory

`libctl inventory`

The inventory command is a subset of the status command in that it returns the status of drives, I/E slots, and tape slots.

load

`libctl load 3 0`

The load command instructs the library to load a tape from the slot listed into a drive. The tape slot is required, but the destination drive may be omitted and the default of the first drive (drive 0) will be assumed.

unload

`libctl unload 0 3`

The unload command instructs the library to pick the tape from the listed drive and return it to the listed slot. You may omit either the slot and libctl will return the tape to the slot it was originally picked from, or

both the drive and the slot and libctl will use drive 0 and the slot the tape was originally picked from.

unload NOTE: Some libraries require that the tape be physically ejected from the tape drive before being instructed to pick it. Use the 'tapectl unload' command to eject the tape from the drive before issuing the 'libctl unload' command.

move

```
libctl move 4 10
```

The move command allows you to relocate a tape from one slot to another within the library. The example command moves the tape in slot 4 to slot 10. Not all libraries support this operation, so check the libctl return code if nothing seems to have occurred. Also, an error will occur if the from slot is empty or the to slot is full.

export

```
libctl export 14 1
```

The export command is only useful if your library has a true Import/Export slot (sometimes called a mailslot or an EE slot). It provides a method of exchanging tapes within the library without requiring that the library be physically opened. This is mainly important in larger libraries where opening the library causes a full reinitialization. This command tells the library to pick the tape from the listed slot and place it into the listed I/E slot. The example picks the tape in slot 14 and places it into the first I/E slot.

import

```
libctl import 1 14
```

The import command is the opposite of the export command. It is only useful if your library has a true Import/Export slot. After you've placed a tape into the I/E slot and locked it, this command will pick the tape and place it into the library's inventory.

unlock

```
libctl unlock
```

The unlock command unlocks the I/E slot. It is not supported by all libraries and can actually cause a system hang between the library and the computer. We recommend that you use your library's control panel for locking and unlocking the I/E slots.

OS X Standard IO Streams - a One Page Primer

Under Mac OS X, reading or writing data on a tape requires either a 'tape aware' application such as TOLIS Group's BRU, or a utility that can read the raw data off of the tape and provide it to the appropriate application or take the output of the application and send it to the tape drive. These applications, such as `tar`, `cpio`, or `pax` provided in OS X are not tape aware. The TOLIS Tape Tools' `taperead` and `tapewrite` utilities provide the ability to read and write raw data with tape drives under OS X.

The `taperead` (and `tapewrite`) utilities make use of the Unix standard streams – standard in (`stdin`) and standard out (`stdout`) – to allow access to tape media by normal tools such as `tar` and `cpio`. The mechanism that is used is called a command pipeline. This mechanism allows the output of one tool to provide the input to the next tool in the pipeline. For example, if you wanted to examine the contents of a file containing the names and email addresses of your friends called `myfriends.txt`, but only display the last names in A-Z sorted order, you could use:

```
cat myfriends.txt | awk '{ print $2 }' | sort
```

There are actually 3 commands issued in that single line: `cat`, `awk`, and `sort`. The output of `cat`, which would normally be displayed on your screen, becomes the input for the `awk` command. The `awk` command then pulls out the second field (the last name for our example) and sends it out to become the input of the `sort` command. The `sort` command then sorts the list of last names provided by the `awk` command's output and displays them in alphabetical order on your screen.

The `taperead` and `tapewrite` utilities use this same method of providing input to or taking output from the various archive utilities. This enables you to either read or write tapes using OS X tools that do not normally have the ability to access tape drives.

Reading a tape with taperead

The taperead utility reads the raw data from a tape and provides it as standard output to be used by whatever utility that you wish to pipe it to. For example, to read a tar tape written on a Sun Solaris system and restore its data onto your OS X system, the following command can be executed in the Terminal:

```
taperead -f ntape0 | tar -xvf -
```

The result is that taperead will read the raw data off of the tape in the ntape0 device and send the data stream to the input of the tar command. The tar command will then process this stream as if it were reading the tape directly and restore the data to your OS X system, matching the original paths stored on the tape.

If you execute taperead without piping its output to another application, you will see the output written to your Terminal session. There is no damage in this, but your Terminal session can become unreadable. To repair this, simply exit the Terminal and start a fresh session.

The taperead command has 3 options, none of which are required unless the defaults are not correct:

```
-b size      (in bytes) - default is 10240  
-f device    - default is ntape0  
-v           - enable verbosity
```

By default, it will use ntape0 as the device with a buffer size of 10k (10240 bytes). This default buffer size was selected because it is the default buffer size used by tar on all systems. Be aware that if this default buffer size was changed when the tape was originally written (using tar's -b option), you will need to specify the appropriate size when reading the tape with taperead and in the tar command. The following examples explain this more completely.

taperead examples with tar

A tar tape is written under Linux using a 64K (65536 bytes) buffer size:

```
tar -cvb 128 -f /dev/nst0 /home/tjones
```

To restore this tape to your OS X system using taperead and tar, you would use:

```
taperead -b 65536 | tar -xvb 128 -f -
```

In the above example, we've told taperead to read the data on the tapes in 64K chunks and send it (pipe it) via our stdout channel to tar's stdin channel (see Pp 13 above). We then tell tar extract (restore/recall) the data (-x) while printing it's status (-v) and to expect 128 512 byte blocks per read (-b 128) and to read from stdin (-f -).

Note that tar measures its buffer size in 512 byte blocks instead of bytes or Kilobytes. Therefore, the 128 passed to the original tar command is the equivalent of $128 * 512$, or 65536 bytes (64k). Also, tar's -f argument is the '-', our stdout.

The ***most important thing that you need to know when receiving media written by someone else*** on a Unix system with tar is what tar blocksize was used (the -b argument) to create the tape. Unless you know the blocksize of the original creator's tar command, it can be nearly impossible to determine once you receive the tape(s). If you have any input into the tape operation, we recommend the following tar blocksize setting on the Unix system creating the tapes (the system creating the tapes should **ALWAYS** use variable blockmode in talking to the tape hardware):

DAT, VXA, DLT, AIT, LTO-1:

```
-b 128
```

SDLT, SAIT, LTO-2 through LTO-4:

```
-b 1024
```

Important tar notes

tar does not monitor the permissions of the user restoring the data, so any user may restore the data from the other system, not just the superuser 'root' or the system administrator account.

tar automatically strips any leading '/' off of its archives, so if you want to restore the data to its original location and the archive was created from the root of the original system, you must cd to '/' (root) on your OS X system and either use sudo or su to the root account to ensure all data and paths are restored as closely to the original state as possible. If you restore to your current path, you will recreate the original directory structure under your current working directory.

taperead examples with cpio

The `cpio` tool is slightly different in the manner in which it is used. The examples below are not designed to be a `cpio` tutorial, but simply to provide examples of how to use `cpio` in conjunction with `taperead`. Please refer to the many sources of information on `cpio` for more specific command descriptions (`'man cpio'` to get started).

To read a tape in drive `ntape1` created under IBM's AIX using the `cpio` defaults:

```
taperead -f ntape1 -b 5120 | cpio -vicdumB
```

This will read the data in 5120 (5K) chunks and pass it through to `cpio` for restore. For information on the `cpio` options used, refer to the `cpio` man page.

Further information on the various tools available is provided via the unix "man pages" and can be read using the man command line tool:

```
man tar  
man cpio  
man pax
```

There is also vast amounts of data available in the form of "HOWTOS" on the web.

Writing a tape with tapewrite

In much the same manner that taperead will read a raw data stream from a tape, tapewrite will accept a raw data stream from another application's standard output and write it onto tape.

Since the default archive utilities are not tape aware under OS X, the TOLIS Tape Tools tapewrite utility provides the mechanism required to enable these commands to write to tape.

An example session using tar to backup the /Users directory under OS X looks like:

```
tar -cvf - /Users | tapewrite
```

Notice that, like in reading a tape, tar's `-f` argument is '-', or stdout in the case of writing an archive.

The same conditions apply to writing archives using tapewrite that we mentioned in our discussion of taperead.

The main thing to keep in mind is that the default I/O buffer used in tapewrite should match the buffer used by the archive utility. Therefore, if you boost tar's buffer size with the `-b` option, you should also provide the appropriate value to tapewrite using the `-b` option:

```
tar -cvb 128 -f - /Users | tapewrite -b 65536
```

Also, because of the raw nature in which tapewrite processes your data, you can pretty much use it with any tool's standard output. While we wouldn't necessarily recommend this use, you could actually combine the `dd` command with tapewrite to create an image of your primary disk drive on tape:

```
dd if=/dev/rdisk0 bs=1024 | tapewrite -f ntape0
```

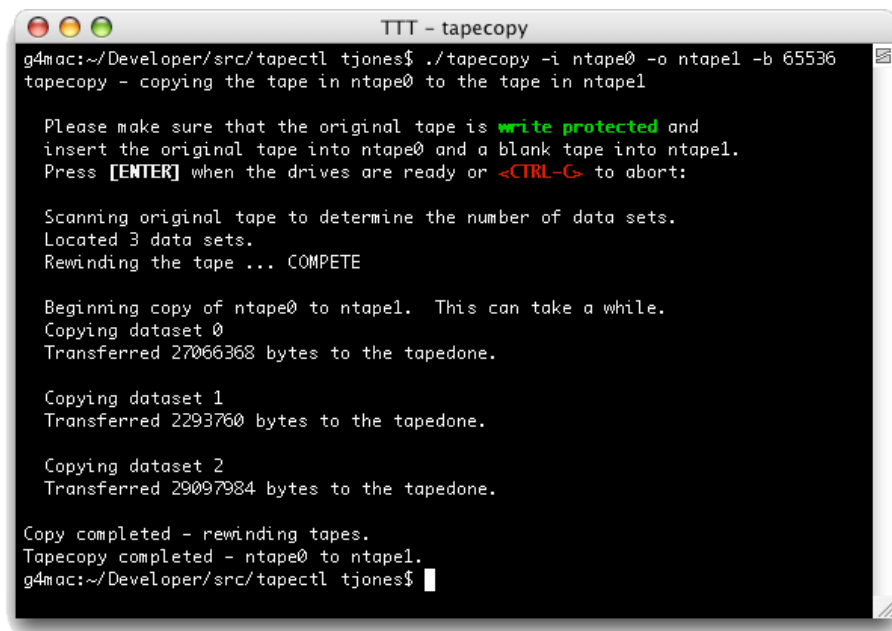
Duplicate Tapes with tapecopy

The tapecopy utility allows you to duplicate tapes in 2 tape drives. The tape in the first listed drive is read and its contents are written in the same format to the tape in the second drive. The two drives do not need to be the same type, only the tape in the second drive must hold all of the contents of the tape in the first drive.

To copy the tape in the first drive (ntape0) to the second drive (ntape1):

```
tapecopy -i ntape0 -o ntape1
```

Once the copy begins, tapecopy will examine the original tape for multiple data sets. It will then copy all recognized data sets (data sets are separated by filemarks) to the new tape as shown below.



```
TTT - tapecopy
g4mac:~/Developer/src/tapectl tjones$ ./tapecopy -i ntape0 -o ntape1 -b 65536
tapecopy - copying the tape in ntape0 to the tape in ntape1

Please make sure that the original tape is write protected and
insert the original tape into ntape0 and a blank tape into ntape1.
Press [ENTER] when the drives are ready or <CTRL-C> to abort:

Scanning original tape to determine the number of data sets.
Located 3 data sets.
Rewinding the tape ... COMPETE

Beginning copy of ntape0 to ntape1. This can take a while.
Copying dataset 0
Transferred 27066368 bytes to the tapedone.

Copying dataset 1
Transferred 2293760 bytes to the tapedone.

Copying dataset 2
Transferred 29097984 bytes to the tapedone.

Copy completed - rewinding tapes.
Tapecopy completed - ntape0 to ntape1.
g4mac:~/Developer/src/tapectl tjones$
```

tapecopy at work on a multi-data set tape

TOLIS Group Support Notes

While TOLIS Group support will assist you with the usage of the TOLIS Tape Tools utilities, please keep in mind that TOLIS Group is not a support team for the use of tar, cpio, pax, or any other non-TOLIS product.

If you uncover other standard Mac OS X tools that can be used in conjunction with the TOLIS Tape Tools, please send notes to the TOLIS Tape Tools mail list and we'll document them for others.

Support for the TOLIS Tape Tools is provided via the online mail list:

`tapetools-user@tolisgroup.com`

To subscribe, send an email to `tapetools-user-join@tolisgroup.com`. The email address that you use to send the email will be subscribed.

You may also visit the BRU Product Knowledge Base for additional Tape Tools support at:

<http://knowledgebase.tolisgroup.com>

To join the TOLIS Tape Tools Announcement list so you may be notified of product updates, send an email to `tapetools-announce-join@tolisgroup.com`. The email address that you use to send the email will be subscribed.

TOLIS Tape Tools License Agreement

TOLIS Group, Inc. **LICENSE AGREEMENT**

This License Agreement ("Agreement"), provided by TOLIS Group, Inc. ("TOLIS"), governs the use of the object code version of the TOLIS Tape Tools brand computer software, documentation and materials accompanying this Agreement or otherwise provided in connection herewith (collectively, "Software"), owned by TOLIS, by the person or entity ("Client") that has clicked on the "Agree" button below.

IF YOU DO NOT AGREE WITH THESE TERMS, YOU MUST SELECT THE "DO NOT AGREE" OPTION AND YOU MUST NOT INSTALL OR USE THE SOFTWARE.

1. LICENSE AND USE RESTRICTIONS.

Subject to all other terms of this Agreement including the payment of any applicable fees, TOLIS hereby grants to Client a non-exclusive, non-transferable license, without the right to grant sublicenses, to use one (1) copy of the Software solely for Client's own, internal purposes. The foregoing license includes the right of Client to make a reasonable number of copies of the computer programs contained in the Software solely for backup and archival purposes; provided, however, that all such copies shall be deemed Software for purposes of this Agreement. The foregoing license shall terminate immediately and without notice for any breach of this Agreement by Client, including any failure to pay fees when due. Upon any such termination, Client shall immediately destroy or delete any and all Software and promptly confirm in writing that Client has done so.

This software is licensed for installation on exactly one (1) computer system per license purchased.

The Software is and shall remain the sole and exclusive confidential and proprietary property of TOLIS, subject to protection under the intellectual property laws of the United States and those throughout the world. Client agrees not to use or disclose the Software, during and after the term of this Agreement, except as expressly permitted by this Agreement. Client further agrees not to modify the Software, remove any notices or markings on the Software, or reverse compile, reverse assemble, reverse engineer or otherwise attempt to learn or disclose the trade secrets contained in the Software, transfer the Software in whole or in part over a network, or permit any third party to do any of the foregoing. Nothing in this Agreement shall be construed as conferring any license under any of TOLIS's intellectual property rights, whether by estoppel, implication, or otherwise, except for those licenses expressly granted herein.

2. WARRANTY AND DISCLAIMER.

TOLIS warrants that for a period of sixty (60) days from the date of receipt by Client of the Software, the media on which the Software was delivered shall be without defects in materials or workmanship. TOLIS agrees to replace any defective media which is returned to TOLIS within the foregoing sixty (60) day period. TOLIS may make available to Client

additional services, including updates, enhancements or improvements of or to the Software, under separate written agreement, and for additional payment.

THE FOREGOING WARRANTY IS THE ONLY WARRANTY GIVEN HEREUNDER. EXCEPT AS OTHERWISE PROVIDED ABOVE, THE SOFTWARE IS PROVIDED ON AN "AS IS" BASIS, WITHOUT ANY WARRANTY WHATSOEVER. ALL EXPRESS, IMPLIED OR STATUTORY CONDITIONS, REPRESENTATIONS AND WARRANTIES, INCLUDING ANY IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, OR NON-INFRINGEMENT, ARE DISCLAIMED. Some states do not allow the disclaimer of implied warranties, so the foregoing limitations may not apply to you.

3. LIMITATION OF LIABILITY.

TOLIS SHALL NOT BE LIABLE FOR ANY INDIRECT, SPECIAL, INCIDENTAL, CONSEQUENTIAL OR EXEMPLARY DAMAGES ARISING UNDER THIS AGREEMENT OR IN CONNECTION WITH THE SOFTWARE, REGARDLESS OF WHETHER ADVISED BEFOREHAND OF THE POSSIBILITY OF SUCH DAMAGES. IN NO EVENT SHALL THE LIABILITY OF TOLIS HEREUNDER EXCEED THE SUM OF ONE HUNDRED DOLLARS (\$100), REGARDLESS OF THE CAUSE OF ACTION, IN TORT, CONTRACT OR OTHERWISE.

4. GENERAL.

Any action related to this Agreement shall be governed by the substantive laws of the State of Arizona, without regard to conflicts of law principles. The State and Federal courts located in Maricopa County, Arizona, shall have sole jurisdiction over any dispute arising hereunder, and the parties hereby consent to the personal jurisdiction of such courts. Neither this Agreement, nor any rights hereunder, may be assigned by operation of law or otherwise, in whole in part, by Client without the prior, written permission of TOLIS. Any sale of more than fifty percent (50%) of the common voting stock of, or other right to control, Client shall be deemed an assignment. Any purported assignment without such permission shall be void. Any waiver of any rights of TOLIS under this Agreement must be in writing, signed by TOLIS, and any such waiver shall not operate as a waiver of any future breach of this Agreement. In the event any portion of this Agreement is found to be illegal or unenforceable, such portion shall be severed from this Agreement, and the remaining terms shall be separately enforced. The parties agree that any breach or threatened breach of this Agreement by Client is likely to cause TOLIS damage that is not fully reparable by payment of damages, and further agree that in such case TOLIS shall be entitled to seek and obtain injunctive or other equitable relief to protect its rights hereunder. Client's performance hereunder and use of the Software shall at all times comply with all applicable laws, rules and regulations, including those governing export of technical information, and Client shall fully indemnify, defend and hold harmless TOLIS against any violation thereof. This Agreement is the entire agreement between the parties with respect to this subject matter, and supersedes any and all prior or contemporaneous, conflicting or additional communications, negotiations or agreements.

Thank you for doing business with TOLIS Group, Inc.!

TOLIS Group, Inc.
8687 E Via de Ventura
Suite 115
Scottsdale, AZ 85258
bruinfo@tolisgroup.com