TransAmiga BBS v1.2

Shareware FidoNet BBS Software for the Amiga

This documentation was revised and created by Patrick Langer

This document was created using texinfo V2.125 and PasTeX V1.4beta6 on 13 June 1996

1 Introduction

Welcome to TransAmiga BBS, a Shareware bulletin board system (BBS) for the Amiga. TransAmiga's aims is to provide an easy to setup and use BBS program for not-so-demanding tasks, and at the same time, incorporate flexibility and advanced features for more serious purposes, such as networking and multi-line.

Installing TransAmiga for the first time is made very easy using the Installer script. All necessary steps are taken to setup the BBS so you can run it without further configuration, though this would be only a very limited setup. Customizing the BBS is too complex to be handled by a script so read this manual to gain full control over the BBS. You will be amazed by the possibilities TransAmiga has.

Please see Section 7.2 [License], page 85 for the usage license.
Some familiarity with AmigaDOS, ARexx and general BBS terms is assumed.

1.1 TransAmiga Roots

TransAmiga was originally written and designed by Timothy Aston, a canadian programmer. He started in 1989using AmigaBasic, and developed it over the years. After switching to HiSoft BASIC, the v1.1 was released at the end of 1991. Developement slowed down after that because Tim had other things in mind than just working on TransAmiga. The Sysop of the German support board, Sami Radwan, offered Tim his help for further developement which was greatly accepted. They worked closely together until the release of v1.11 in 1993. At the end of 1993, Sami purchased the whole rights for TransAmiga BBS from Tim and continued the work he already had started a while ago. Development really got a push now and many new features were included, development is still continuing.

1.2 Philosophies Behind TransAmiga

When Tim wrote TransAmiga, he had only ever used BBS software from the caller's point of view. He had never run his own BBS, nor even looked at any BBS software from the SysOp's end. One of his primary objectives when he started it was to create an attractive user interface for the caller. Though much effort has also been made to include in TransAmiga a large array of powerful features for the SysOp, it is still a basic idea that the purpose of a BBS is to serve its callers - afterall, a BBS that doesn't get any callers isn't much fun.

Another major philosophy behind TransAmiga is the belief that running a BBS should be fun. The world of Amiga BBS software has become awfully intense and competitive lately, and the prices of high-quality BBS software averages well over $100. Both these facts appear contradictory to the idea that running a BBS is supposed to be a hobby. TransAmiga tries to rekindle this spirit of fun that started the BBS world in the first place by not always trying to be "one up on everyone else", but by providing a good quality software package at a reasonable cost.

Lastly, TransAmiga relies heavily on its user-base of SysOps and contributions from TransAmiga users are valued highly. TransAmiga SysOps have done an excellent job in creating tools and utilities and due to it's powerful Arexx interface, which became a very important part of TransAmiga, it is very easy to create own tools or convert tools from other BBS packages.

1.3 System Requirements

TransAmiga BBS requires at least an Amiga 500 with 1 MB of RAM and two floppy disk drives running OS1.3. Though the BBS will function fine in this environment, it was designed for more powerful systems. Additional memory and a hard drive of at least 100 MB are highly recommended. You will also find an accelerated CPU and OS2.x or above advantageous to have when running TransAmiga. At least one function needs at least OS2.x and definetly doesn't work with older OS versions (ScreenToFront function). It is strongly recommended to upgrade to at least OS2.04 as support for older OS versions cannot be guaranteed for future releases.

The following chart shows TransAmiga's minimum requirements, the kind of equipment that is recommended for good use of the BBS, and what you will find ideal for running TransAmiga:

Item	Minimum	Recommended	Ideal
System	Amiga 500	Amiga 2000	Amiga 4000
CPU	68000	68030	68040
RAM	1 MB	3.0 MB	6.0 MB or more
O.S.	1.3	2.0	3.1
Modem	1200 baud	14400 baud	V.Everything
Serial Port	built-in port	built-in port	Multi-port card
Drive Space	2 floppies	100 MB	500 MB or more

1.4 Features

Here is a brief listing of the features of TransAmiga BBS:

- General
  - Configure size, depth, and font of TransAmiga's screen.
  - Basic Iconify support
  - Full ARexx support with over 150 ARexx commands. ARexx macros can be installed almost anywhere to provide maximum flexibility.
  - Can run many SkyLine/C-Net ARexx macros with little modification.
  - Arbitrates access to all BBS files that allows several copies of the program to be run concurrently without the risk of access collisions, allowing a multi-line BBS system to be run.
  - Simple "idiot-proof" setup program.
  - ANSI colour graphics optional for each user, with local ANSI emulation (including both Amiga and IBM ANSI sequences, as well as full 16 colour ANSI).
  - RIPscrip supported (though not fully implemented in v1.2)
  - Optional command stacking or hot keys, selected by each user.
  - When entering commands in the BBS, users have the full range of command line editing keys available, as well as command line history.
  - Most displays and processes can be interrupted by the user
  - Intuitive user interface.
  - Some executables optimized for >680x0 and FPU
- File Libraries
  - Multiple file areas, each with a minimum access level, defined by the SysOp.
  - File transfers using external protocols setup by the SysOp.
  - Optional upload:download ratio for each user (size or number of files dependent)
  - Batch and non-batch protocols supported.
  - Files can have an optional long description.
  - When listing files, users have a choice of either viewing them a page at a time, or one file at a time.
  - Users can mark/unmark files for a batch download.
  - FILE_ID.DIZ supported
  - CD-ROM usage supported (for v1.2, only Aminet CD)
- Message Conferences
  - Multiple SysOp defined message areas, each with a minimum access level.

- Messages can be public to be viewed by all, or privileged to be read only by the addresser, the addressee, and those with privileged access.
- User can check for mail addressed to him/her at logon.
- Message area can be defined as either local, private, netmail, or echomail.

– FidoNet
- Able to read and write FidoNet messages according to the FTS-0001 standard.
- Uses existing FidoNet software that has been time tested and proven reliable. Unlike other BBS programs it does not require it's own custom FidoNet utilities, so no battling with bugs and unimplemented features.
- Works with most popular Amiga FidoNet utilities, including TrapDoor, GMS, Foozle, MailManager, Areafix, AmigaTick, etc.
- You can set a different 5-D address for each FidoNet message conference.
- Able to read the Foozle message format.
- Can create crashmail and file attaches.
- Nodelist support.
- Allows you to run mail processing software while users are online.
- Supportive of 4-D mail processing software, and can be run in either a point or a node environment.

– SysOp Controls
- Optional daily time limit for each user.
- User can leave a one line comment to the next user when logging off.
- Bulletin menu for posting text files. Each bulletin requires a minimum access level to view.
- Every command can be assigned to any key, and can be given any name.
- Every command has a minimum access level.
- Any command, file area, message area, or bulletin for which a user does not have sufficient access, is invisible to that user.
- 256 different access levels plus a set of 8 access flags, for controlling access to almost every facet of the BBS.
- SysOp can view and edit a user's stats from within the program while the user is online.
- Fully interactive remote shell.
- Supports three different kinds of editor: a line-editor for ASCII callers, a full-screen editor for ANSI callers, and the editor of your choice for local calls.
- User maintenance utility that can be used from the BBS as a door or from the CLI.

– Configurability
- You set the key, access, flags, and text for every command in the BBS.

- Add your own commands through ARexx.
- You set the colours that should be used for text displays within the BBS, so the BBS looks like you want it to.
- Through special control codes, text files can be made to display over 70 different pieces of system/user information.
- Create your own custom menus and menu prompts, or have the BBS displays it's own default style menus.
- Fully configurable screen setup including screen size, depth, font, mode, etc.
- custom user-definable pulldown-menus

1.5 Registered vs. Unregistered

TransAmiga is a shareware package, meaning you get to look at it free of charge, and once "evaluating" becomes "using", you are obliged to register. This method allows you to "try before you buy", to get a really close look at it before you decide to pay.

You may use the unregistered version of TransAmiga for up to 20 days before you register. Near the end of this manual, full details are given on how to register, and what you get in return, so be sure to follow that carefully.

If you have not registered TransAmiga yet then TransAmiga will still work fully, however, users will not be allowed online for more than 20 minutes in a day. Additionally, the Iconify option doesn't work. Also, you cannot use the full screen online editor option.

When you register, you will get a small personalized key file that will remove this restriction from TransAmiga. The actual TransAmiga program itself is always the same, it is the presence of the key file that determines the difference between unregistered and registered.

2 Setting Up

This part mainly describes how to set up the TransAmiga BBS. Thanks to Installer, most of this can be done via the Installer script, but nevertheless, some things are too complicated to be handled by a script. You have to do that by hand.

If you are updating from an older version, you have to use the update option of the Installer script as there were so many changes in the configuration files that this is handled by a special command to insert the new config lines into your existing configuration.

Don't be worried: the Installer script gives you the possibility to backup everything in '`bin/`' and '`configs/`'. Be careful with '`rexx/`' if you have customized standard scripts in there, they could be overwritten.

2.1 Preparation

If you got the TransAmiga package in an archive, just unpack all contents to a seperate directory. If you have enough free memory (about 2,5 MB), you can also use your ram disk, as all necessary stuff will be copied to a seperate directory.

To start the installation or update, just click on the Install icon. The Installer will then take you through all the necessary proceedings. If you first want to take a look at the things the Installer will do, choose 'pretend to install' at the beginning, although the Install script will not make weird changes to your system configuration.

2.2 The Modem

TransAmiga should be able to work with virtually any kind of modem, because you set all the modem commands by yourself.

For TransAmiga to work properly with your modem, it must be set to include the baud rate on connection messages in the format CONNECT XXX, where XXX is the baud rate unless it is a 300 baud connection in which just CONNECT should be sent, eg. CONNECT 2400. On most Hayes compatible modems, this is the default action, but on some you must enter the command ATX4, which you would insert in you modem initialization string (read on for more info on your modem initialization string). Consult your modem manual for information on its commands.

It is very important that your modem is set so that Carrier Detect (CD) and Data Terminal Ready (DTR) are not forced. On some modems, you will need to adjust some DIP switches. On modems with Non-volatile RAM, you may need to enter:

```
AT&F
AT&D2&C1&W
```

from a terminal program. Again, consult your modem manual to find out more about this. ZyXEL modem users should change their modem init string: `~~+++~~~` should be the start of the string if not allready. Most ZyXEL ROM versions don't handle DTR right (&D2). TransAmiga also requires a 7-wire serial cable to work (most serial cables are wired as such, this is rarely a problem).

TransAmiga has been fully tested with many modems like the Courier HST DS or ZyXEL. TransAmiga supports the ability to lock the serial port baud rate for maximum throughput. How to do this is explained later on.

2.3 Directories

This section will describe briefly the directories TransAmiga uses. Most of them are created for you by the Installer script. This is just to give you an idea of the directory tree and how it is used by TransAmiga.

TransAmiga uses several directories which should be assigned, though this is not necessary but recommended:

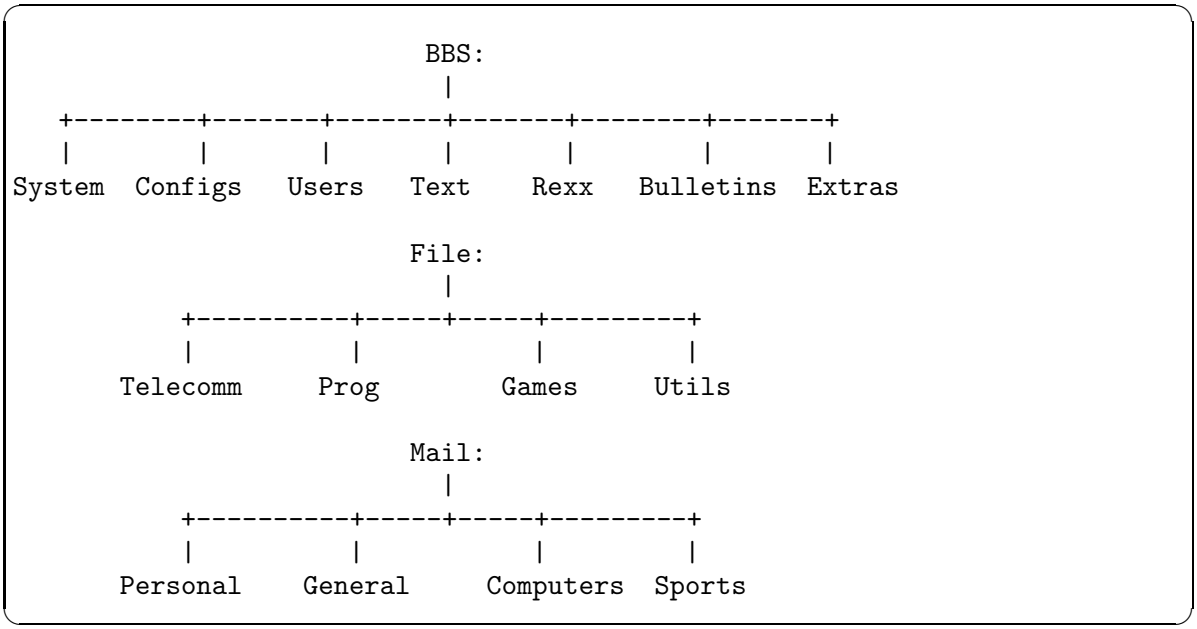
`BBS`	The main directory containing all BBS related files and directories.
`Mail`	This directory contains all subdirectories for the message areas
`File`	This directory contains all subdirectories for the file areas.

Disk usage can vary very much depending on your BBS' intentions. If you just want to use it as a private mailbox, it can fit onto a usual floppy disk. If you want to offer nets like FidoNet or the like, you may need about 50-100MB disk space or more. And if you want to install a rather big file base, there can never be enough disk space, minimum 500MB.

Within the `BBS`, `File`, and `Mail` directories, TransAmiga uses several other directories:

- The bin directory for keeping all the executables for use with the BBS and should be in your search path (‘`BBS:Bin/`’)
- The system directory for keeping various general files for system upkeep (‘`BBS:System/`’)
- A configurations directory for keeping the various files that say how the BBS is set up (‘`BBS:Configs/`’)
- The users directory for keeping individual user files (‘`BBS:Users/`’)
- A text directory where all help files, menus, title screens, and other text files shown to users are kept (‘`BBS:Text/`’)
- The bulletins directory where file to be displayed at the bulletins menu are kept (‘`BBS:Bulletins/`’)
- The extras directory for miscellaneous files created by online games, BBS utilities, and other external BBS related tools (‘`BBS:Extras/`’)
- The ARexx macros directory, where the BBS looks for any ARexx macros. If you do not intend to run ARexx, then you will of course not have an ARexx directory (‘`BBS:Rexx/`’)
- One directory for each file and message area you choose to setup. These naturally contain all the messages and files for those areas.

These simple tree diagrams illustrates an example of how your directory structures may look with example file and message areas:



2.4 TransAmiga Configuration Tool

TransCfg is a utility used to help you to make setting up and maintaining the BBS easier. If you are installing TransAmiga using the supplied Installer script, you can skip the first part dealing with the master configuration, as the script will ask you for all needed values.

By default it will be located in your '`BBS:Bin/`' directory, although it doesn't particularly matter where you put it.

TransCfg must be loaded from the Shell. Upon execution, it will ask you for the name of TransAmiga's master configuration file. The master system configuration file contains general information that is vital to the operation of the BBS. You can call this anything you want, but it is recommended that you call it TransAmiga.Cfg, and place it in your '`BBS:Configs/`' directory. Therefore, you would enter '`BBS:Configs/TransAmiga.Cfg`'.

When you give TransCfg the name of your master configuration file, it will see that this is your first time setting up the BBS. What follows will be a series of questions that you must answer in order to get the BBS setup according to how you want it. Before each set of questions there is a brief paragraph explaining what you are being asked for.

If you wish to set up a system running multiple lines, you'll have to be a bit patient. Just set up a single line for now, adding additional lines will be described later on (see Section 3.11 [TransAmiga Multi-Line], page 31).

When entering information, default values will be shown in [] , you may accept the default by just hitting `RETURN`, or you can enter your own. Default values will automatically be set for you. If you followed the suggestions for directory names and assignments given above, then you will be able to accept most of the defaults as they are.

2.4.1 General

This is where you setup your master configuration file. There will be several questions covering the general setup of the BBS. Most questions are easy to answer so they won't be described in full length here. Nevertheless, some questions still need some additional informations to be answered correctly. They will be discribed now:

Loglevel will control how detailed log output is. Valid loglevel range is 1-11. If the entry is lower or higher it gets adjusted to the minimum/maximum. Don't use numbers higher 254! Suggested

loglevel is 6. Higher loglevels may slow down the BBS dramatically as they write very much to the logfile.

For details on the loglevel values, see Section B.7 [TransCfg], page 107.

If custom screen is chosen,opening a backdrop window will display a blue backdrop behind the terminal and status windows. May be used when terminal window dimensions are smaller than the whole screen. Takes up a significant amount of memory. (Blue backdrop may be removed!)

Choosing sizing gadget on the terminal window will enable you to resize the terminal window. NOTE: this will slow down active display when online *a lot*!

The screen/window defaults are for an 8 colour medium-resolution screen that is slightly overscanned so as to fit 80 columns and 23 rows.

The file you specified when you executed TransCfg will now be created with all the information you just entered in it.

2.4.2 Messages

Next you will set up the message areas. The purpose of having separate message areas is so that completely different topics will be separated. You may setup as many message areas as you wish. The defaults are for just two simple message areas, one for private mail and one for public mail.

The first few questions here deal with FidoNet, which will not be described just yet, so skip over these for now by hitting `RETURN` until you are asked how many message areas to setup. If you are interested in FidoNet, it will be covered in detail later on (see Chapter 5 [Fidonetting], page 47).

For each message area, you must choose an appropriate name. Examples might be "Sports Talk", "Current Affairs", "Amiga Users", etc. Next, enter the full path to that message area's unique directory. If you haven't made the directory, TransCfg will make it for you, though any parent directories must already exist. It is recommended that you have your message directories within '`Mail:`', but this is not essential. Spreading message areas over multiple drives is perfectly acceptable, albeit a little confusing.

Now enter the type of message area. For the time being, enter 1 for a normal message area or 0 for an all-privileged message area (that is, all messages entered will automatically be assumed

to be private), see the section on FidoNet later on in this manual for information on net and echo mail areas (see Chapter 5 [Fidonetting], page 47). Each area has a minimum access level, and also flags associated with it. For more information about access levels and flags, see Section 4.4 [Access Levels], page 43.

Each area also has minimum access levels for reading and writing. These can be used to create read-only message areas, or to keep first time callers from entering messages in certain areas.

For each area you must specify a quote lead-in line. When users are quoting lines from the message they are replying to, it is often desirable to have a simple intro line before the quotation so that people reading the message know what's going on. Simply enter the text for this line. You may enter a few codes to have certain information substituted into the string:

`%n`	the name of the user who wrote the message you are replying to
`%a`	the name of the person that message was addressed to
`%d`	the date the original was written
`%t`	the time the original was written

Some example quote lead-in lines:

```
In a message sent <%d %t>, %n wrote to %a:
%n uttered this gibberish at %t, %d to %a:
```

You may decide whether or not you wish to allow users to post messages under their "handles" in each conference. If you say yes, users that have setup a handle for themselves will be prompted as to whether or not they wish to use it for the message they are posting.

You shouldn't normally need to adjust the low and high message number, and for now leave them at zero. For all local message areas you must enter the maximum number of messages to keep in the area at one time. Once that limit has been exceeded, TransAmiga automatically deletes messages, starting at the oldest. Finally, you are prompted to reset the users' last read pointers. You must answer Yes to this option whenever you are first setting up an area, as this initializes the data file TransAmiga uses to keep track of users' last read messages.

After setting up the individual areas, some other questions must be answered. You must enter the user name messages entered via the feedback command should go to, and to which area number they should be placed in.

Finally, you will be asked for the external editor commands. Once again, there are some % codes you can use to substitute in information:

`%n`	the number of the BBS line starting the editor
`%f`	the filename of the file that should contain the message text to be saved.
`%q`	the name of the file containing the quoted text when you are replying to a message.
`%b`	the real (non-locked) baud rate of the online caller.

The full-screen editor command is the program that gets run when online callers invokes the full-screen editor (see Section 3.8 [Text Editors], page 25 for a description of the different text editors). You should use the provided TAEd program for this (registered users only). The command need simply by:

```
BBS:Bin/TAEd %f -q%f -n%n
```

See Section B.1 [TAEd], page 103 for more details on commandline usage.

For the local editor command, you should give the command to start up your favourite text editor. Any editor should work, such as Ed, MEmacs, CEd, etc. For example, to use CEd as your text editor, you could use:

```
Work:Tools/CEd %f -keepio
```

Most other text editors should require similar command lines (as CEd is self-detaching, you have to prevent it doing so with `-keepio`, otherwise TransAmiga would continue as soon as Ced is started).

2.4.3 Files

Like the message conferences, the files section is divided up into separated libraries. The defaults are quite simple, just two file areas. If you are using a dual floppy drive system, then you will not have enough space for file libraries, but you will still have to setup a "dummy" area. Just call this anything, have the directory as '`RAM:`', and give it access 256 so no one can get there.

Prior to setting up the libraries, you must specify what is considered to be a "full" drive. TransAmiga will refuse to allow uploads when less than the amount you specify is free. Generally

it is desirable to set this to at least 50k so that the drive will not become full from messages being posted and other BBS operations.

Setting up the actual file libraries setup to start with is quite similar to the message area setup. Each file library must have a name, a unique path, access, flags, and read and write access levels. It is recommended that all your file directories are somewhere within '`File:`', but as with message directories, this is not essential.

After this, you must setup the file transfer protocols. TransAmiga executes external AmigaDOS programs to do file transfers. This has been done to allow for maximum flexibility when it comes to setting up file transfer protocols. For most of your protocols, it is recommended that you use the included TrXPR program, as it was written with TransAmiga specifically in mind, and supports the popular XPR libraries. The defaults will give you the settings for most protocols via TrXPR.

See Section B.18 [TrXPR], page 114 for details.

For each protocol, you must enter the name. Then you must say whether or not the protocol is capable of doing batch transfers (where multiple files can be sent in one session). Lastly, you enter the send and receive commands respectively. TransAmiga allows a few imbedded commands for automatically substituting values into the command line:

`%n`	the number of the BBS line
`%d`	name of the serial device
`%u`	serial device unit
`%f`	the file to be transferred (or directory in the case of a batch transfer)
`%l`	the name of the file containing the list of files to transfer (created by the BBS for sends)
`%b`	baud rate of the current caller (the real baud rate, not the locked one, so if you have the serial port locked, don't use this)

For the time being, just accept TransCfg's defaults as they will be sufficient in most cases. See Section 3.9 [File Transfers], page 27 for more information about transfer protocols.

After setting up the transfer protocols, you will be asked what formats you wish to support for archive viewing. TransAmiga allows users to view the contents of an archive in the file areas. You can support as many different archiving methods as you want. Even if you do not wish to support archive viewing, you must setup at least one dummy method. The default is just for viewing LHA archives, but you can always allow for more, such as Zoo, Arc, etc. As an example, say you only

wanted to support viewing of Zoo archives. The name of the archive type you would enter would be 'Zoo'. For the archive suffix, you would enter '.Zoo' because all Zoo files have that suffix, and that's how TransAmiga will recognize that a file is a Zoo archive. The command would be the full path name for Zoo on your disk plus an 'l', which is Zoo's command to look at the contents of an archive. It is also possible to use such programs as Juergen Hermann's '`XArc.rexx`', that automatically recognize the archive type. For these sorts of programs, do not enter a file extension.

2.4.4 Bulletins

You must set up at least one bulletin. Simply enter the file name of each of the text files you wish to display as a bulletin. Do not enter the full path name, TransAmiga looks for all bulletins in your bulletins directory. The default is for one Bulletin called SystemInfo, where you could just say a few lines about what system is being used to run the BBS. Note that the file name of the bulletin doesn't necessarily have to be the title that the user sees. How to edit that is described later.

You must be careful when setting up bulletins, because each bulletin you set up will require its own entry in the Bulletins.Cmds command file.

See Section 4.1 [Changing Commands], page 33 for more details.

2.4.5 Resetting User File

After these files have been setup, you will see TransCfg's main menu. The last thing you must do is to reset the user file, and to create the first member of the BBS, the SysOp. You will be asked some questions about yourself in order to create the SysOp account, then the user file will be made.

2.5 Limits

TransAmiga has few built in limitations. Theoretically, you should be able to set up 32 000 file areas, message areas, transfer protocols, archive viewing methods, ARexx commands per menu, and bulletins.

The number of messages allowed in a message area is virtually unlimited, however, TransAmiga will not be able to keep accurate last read pointers for users that have read past message 32768. The number of files allowed in an area is limited only by disk space

3 Using TransAmiga

3.1 Running TransAmiga

TransAmiga must be started from Shell or Workbench via the Release 2 Workbench Execute Cmd menu option. Its Shell usage is:

```
TransAmiga [-c<config file>] [-b<baud rate>] [-l<locked baud rate>] [-t<max time>]
           [-n<line>] [-x] [-r] [-s] [-0] [-m<modem connect message>] [-h] [-p]
```

Parameters in [] are optional, and words in <> are descriptors for what you would put in there. Here is a description of the parameters:

`-c`*config file*
: This gives the name of the master system configuration file you want TransAmiga to use that TransCfg created. If it is not given, '`BBS:Configs/TransAmiga.Cfg`' will be used. TransAmiga will not start up if it cannot find all its configuration files.

`-b`*baud rate*
: This one is optional. If specified, TransAmiga will open up the serial port at the supplied baud rate and immediately go to the logon sequence. This parameter is used if there is already a caller online before TransAmiga is started. This parameter is only really useful in conjunction with the `-x` option.

`-l`*locked baud rate*
: This parameter is also optional, and tells TransAmiga to lock the serial port at the supplied baud rate and do not adjust it. This is useful for HST users, as locking the baud rate allows maximum throughput with those types of modems. When used in conjunction with the above parameter, it will over-ride that baud rate, but will use it as the "real" baud rate that the connection is actually at.

`-t`*max time*
: Again, an optional parameter. When this parameter is used, then the online time available of the next caller will be adjusted so that it is no more than this value. Also, no time bonus will be given after uploads. This parameter is only useful in conjunction with the `-b` and the `-x` options.

`-n`*line* gives the BBS line number. If not given, TransAmiga assumes line number 0. This is only required when you wish to run TransAmiga several times simultaneously. Valid line numbers range from 0 to 99.

`-x` Tells TransAmiga to exit after the next caller. This useful as it allows you to perform system maintenance tasks between calls with out the BBS interfering.

`-r` Tells TransAmiga to wait on its ARexx port after starting up, and not logon any users. This is particularly useful when having a frontend mailer answering the phone, as it allows nearly instantaneous transition from the mailer to the BBS, instead of having to wait for the BBS to load for every caller.

`-s` Tells TransAmiga to open the status window as soon as it loads up. Normally you have to select an option from the pull-down menu to open up the status window. Opening the status window takes up a little more precious memory.

`-0` Switches of carrier check. Useful for nullmodem lines as you no longer need a cable that supports carrier detection.

`-m<`*modem connect message*`>`
Does the same as the ARexx command MDMCONNECT but can be used in the command line. Only use this if you start TransAmiga from TrapDoor.

`-h` Executes ScreenToBack as soon as the screen and windows have been set up. If you use TransAmiga in workbench mode you have to specify `-p` also if you want this to work.

`-p` TransAmiga will grab the ScreenPtr of the screen it's started on (if it's a public screen). Is needed for SCREENTOFRONT ARexx command to work in any case. NOTE! -p cannot be used with <OS2.x!

The most simple way to start up TransAmiga is:

```
TransAmiga
```

If your configuration file is something other than '`BBS:Configs/TransAmiga.Cfg`', then you will have to pass that:

```
TransAmiga -cTransAmiga:Cfg/Main.Cfg
```

In an automated environment, you will probably want the BBS to exit after every call so that you can check the time and see if any system events need to be run, then you'd use:

```
TransAmiga -cBBS:Configs/TransAmiga.Cfg -x
```

HST users will want to lock the baud rate of the serial port. So they would start the BBS up like:

```
TransAmiga -cBBS:Configs/TransAmiga.Cfg -l19200 -x
```

Parameters can be provided in any order and are not case sensitive.

When TransAmiga is started, it will open its window on either the Workbench screen or a custom screen depending on how you set things up. This window is the Terminal window and all BBS I/O is displayed here. You can also open up the status window from either the pull-down menu or using the `-s` command line option, which is used to display information about the current online caller. The window title is used to give you some informations on what's happening during startup or online sessions.

3.2 Troubleshooting

If TransAmiga does not come up and say that it is waiting for a call when you try to start it up, something is set up wrong.

TransAmiga will give you an error message which will hopefully be enough for you to figure out what is wrong. If it's a problem in either a config file or a command file, it will tell you which file has the problem, then you will have to either load the file up in TransCfg or in a text editor as the case may be. Probably the most common problem is not adding command entries to the Bulletin.Cmds file for bulletins that you created in TransCfg.

If a requester or even worse, an alert appears, then there may be a more serious problem. See Tech.doc for some of the error codes.

If you run into problems which can reproduced, use a tool like SnoopDos to try to find out where the problem is. In many cases TransAmiga cannot find a file or something like that. This can be checked with a tool like SnoopDos. If you have no such tool, check the support boards as they most probably have such tools around.

3.3 Pull-Down Menus

TransAmiga has two standard Amiga pull-down menus, "Project", and "Online" and one user definable, "User".

Attention! Don't use a menu item starting an ARexx script (like chat or items of the custom menu) when there is an ARexx script already running or the machine will lock up!

3.3.1 The Project Menu

Most items of this menu are only available when waiting for a logon and are completely disabled when TransAmiga is started in ARexx mode an no caller is online.

`Terminal Mode`
: Enter Terminal Mode. This is a rather simple terminal originally implemented for testing purposes (may be removed!)

`Local Logon`
: Do a local logon. This gives you the option to log into your BBS without the need of calling from outside. That way you can see how your BBS looks like to the users though colours on the local display may differ to the actual colors due to your screen settings.

`Reset Modem`
: Sends the initialization string to the modem

`Status Window...`
: opens the status window showing informations about the current caller online

`About...` Gives some general information

`Quit TransAmiga`
: Quits TransAmiga. Works only with no caller online

`Sysop Logon`
: Do a local logon as sysop. Skips the name & password questions and uses the SYSOP account settings

`Reload Config`
: Closes the window(s) [/screen], frees some memory and starts the setup with reading configfiles. This does not work when a user is online.

`Lock TransAmiga`
: Opens a window asking for a password. ATTENTION! Don't forget the password or you will have to reset to reaccess the line. As soon as you activated this function the menus will be disabled except the Lock TransAmiga function. This function will display a checkmark. Select it again and the unlock window will come up. You have to enter the password for unlocking. The password is *not* case sensitive.

`Iconify` Iconifies TransAmiga

3.3.2 The Online Menu

This menu is only available when a caller is online.

`Start Chat`
: Starts either the internal chat or CChat.trans, if existing

`Stop Chat` Stops the chat

`Log User Off`
: Logs user immediatly off, giving him a short notice that he has been logged off by the SysOp

`Edit Access...`
: Opens a window for editing the current users' access settings

`Edit Info...`
: Opens a window for editing the current users' other settings

3.3.3 The User Menu

This menu is only added if the file '`custom.menu`' exists in the config directory.

'`custom.menu`' has a special format. The first line contains the number of entries you wish to have in this menu. Every following line represents an entry to the menu. The first 19 characters are the name that is displayed in the menu. Then a space must follow. The rest of the line is the name of the ARexx macro that should be executed. The ARexx macro has to be located in the rexx directory of TransAmiga. E.g. you could use trshell.trans there. Check the delivered Custom.menu to see how it works.

Attention! After the last entry there should be a blank line!.
Otherwise TransAmiga may not initialize correctly and exit with a cryptic error message.

This is the enclosed '`custom.menu`':

```
1
Toggle display mode Localdisplay.trans
******************* *******************
12345678901234567890123456789012345678901
         1         2         3
```

The last three lines are not part of the file and are just displayed to make the format more obvious.

3.4 TransAmiga Online

After TransAmiga starts, it must do some initialization. It must load the config files, and the command files, etc.

Depending on how you started TransAmiga, it will either wait for a caller, wait for an ARexx message, or immediately attempt to logon an already online user. If it is waiting for a call you can logon yourself by selecting "Local Logon" from the menu strip. You can also use "Sysop Logon" to avoid the name and password check locally (does not work in ARexx mode). If TransAmiga is waiting for an ARexx message, you can logon locally using an ARexx script utilizing the `LOCAL` ARexx command.

If you are unregistered, a requester will come up as soon as the BBS starts giving the copyright notice. After delay, you can just click the mouse button to make it disappear.

3.5 Logging On

Before anything can be done, a user must logon. You can do this from the machine running TransAmiga by selecting the Local Logon item from the pull-down menu. The logon procedure will first ask you to enter your user name. If this is your first time running TransAmiga, enter the SysOp user name and password that you set up during installation. If a new user calls and wishes to logon to your system, he/she should enter His/her name at this prompt. TransAmiga will see that they do not have an account on the BBS, and will take them through the new user procedure. Users that have set a handle for themselves (if handles are allowed) may also use their handle to logon.

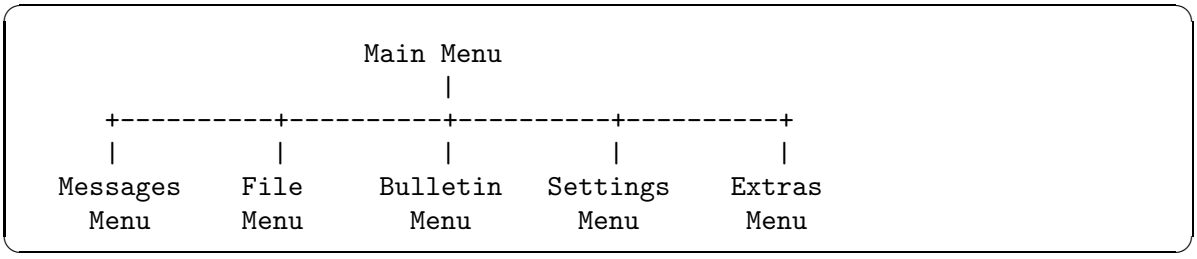
After entering the user name, and if the user is not a first time caller, a password must be entered. You get three tries at the password, and what you type will be hidden so that onlookers will not be able to determine your password.

No spaces are allowed in the password and the username must have at least one space in it. Handles and usernames are limited to chr(32)-chr(126). Additional check for : and / is made to prevent hacks. The maximum length for the username is 25 characters.

The user is logged off if while logging on he enters `RETURN` 15 times at the name prompt. There is no comment displayed.

3.6 BBS Menus

At last, the Main Menu is presented. As with most BBS's, in TransAmiga the user performs functions by making selections from menus. Sometimes making a selection from a menu will take you to another menu. In this way, the menu structure is similar to a tree. Here's how the tree branches out in TransAmiga:



Here is a summary of all the menus in TransAmiga. To get a more detailed description of the commands available, look through the supplied Help files in the text directory.

`Main Menu` From here the user can get to the other menus available from the system. These menus are the Change Setup Menu, the Message Menu, the File Menu, the Bulletin Menu, and the Extras Menu.

`Message Menu`
This is where users can read and write messages to and from other users. Message areas can be set up as either traditional local message areas, FidoNet matrix mail areas, or FidoNet echomail areas. Messages can be either public which can be viewed by all, or privileged which may only be read by the person who wrote the message, the person who the message is sent to, and those with special access. The Message Menu is divided up into multiple message areas. Each area is isolated from the others and requires a certain SysOp defined minimum access to get to.

`Files Menu`
The file areas work in a fashion similar to the message areas in that each area is isolated from the others, and requires a certain minimum access. Here is where users can send and receive files to and from the BBS. Each user has his own Upload:Download ratio, which means that for every specified number of downloads the user must upload a file.

This can now also be dependent on bytes (size of files). If you do not wish to implement this, it can be side-stepped easily.

`Bulletins Menu`
: This menu is for posting text files that users can read. Things you might want to post include BBS listing for the area, rules of the BBS, online game high scores, etc.

`Settings Menu`
: This is where the user can enter information about his/herself, and also adjust some global settings for BBS use, such as password, screen length, etc.

`Extras Menu`
: As it says, this is just an extra menu. You would use this to add ARexx commands to, such as online games.

3.7 Controlling the BBS

TransAmiga gives users two modes of entering commands at menus: command-stacking and hotkeyed commands.

Hotkeyed commands (or just hotkeys) means that the command is processed the second the user presses the corresponding key, `RETURN` does not have to be pressed. This of course cannot be the case when text the user is entering is of variable length, for example, a message conference number or their street address, so `RETURN` must be pressed at the end of the line.

Command stacking allows the user to enter several commands on one line, one immediately after the other. For example, typing in "c?" at the Main Menu will bring take you to the Change Setup Menu (c) and display the available commands (?). You can stack as many commands as you want on a single line. In cases where you'd normally have to press `RETURN`, you can instead enter a space, eg. `M12 N` from the Main Menu will take you the Message Menu (`M`), go to conference 12 (`12`, followed by space because you would need to normally press `RETURN` at this point), and read new messages (`N`).

TransAmiga offers special command line editing features to those with ANSI turned on. You may use the backspace, the delete key, and the left and right cursor keys to edit the line you are entering, just like in the CLI Shell. Additionally, you may call back commands you've previously entered and edit them by pressing `CTRL-K`, and to simply clear any text you typed on the current line, press `CTRL-X` (NOTE: These last two work with ANSI turned off as well). ANSI cursor up and cursor down can be used also to walk through the history buffer (20 lines).

Most of the processes and displays of TransAmiga can be interrupted using `CTRL-C`.

When the user is inactive for a given time (can be set in the master configuration), he will be logged off. 30 seconds before idle timeout TransAmiga sends CHR$(7) to the serial port. This only works while waiting for input. This is not active for local logons.

3.8 BBS Text Editors

At various times when using the BBS, a user will wish to enter a relatively large amount of text, for example, when entering a message, or leaving a long description for a file. To make the entering of text as simple as possible, TransAmiga provides a variety of text editor options for users. There are actually three different sets of editors that can be used, depending on the circumstances: the built in line editor; the external ANSI full-screen editor, called TAEd; and a local editor of your choice.

3.8.1 The Line Editor:

The built in line editor, called TransEd is the simplest type of editor, included for mainly for those without ANSI graphics capabilities. A user enters text continually. At the end of a line there is no need to press `RETURN` as TransEd will wrap words automatically.

In the line editor version of TransEd you can not cursor back to edit something you may have entered incorrectly, you must use the backspace key and delete everything you have entered since the mistake. You may only backspace as far as the beginning of your current line, you cannot go back to a previous line.

Several commands are available in TransEd and are accessed by preceding them with the backslash '\' character and must be the first thing entered on a new line, followed by `RETURN`. Entering a \ by itself will take you to the edit menu where you will be able to get a menu of all the available commands. There are commands to view what you have entered so far, to save your text, to abort what you were doing, to change a string on a line of text, etc.

3.8.2 The Full-Screen Editor

The full-screen editor, TAEd, provides much more power than the line editor. You can use the cursor keys to move all around the text being edited. Attempting to type where there is already

text, causes everything on the line to be moved forward a space to make room for what you are typing. It acts much more like a conventional editor that you would use. The added power comes at the cost of compatibility, as TAEd must rely heavily on ANSI control sequences, so only users that have ANSI terminals may use TAEd.

Most commands understood by TAEd are combinations of either holding Control and pressing a key, or pressing Escape, then pressing another key. For example, to view the list of commands, press the Escape key, then press ?. To save what you are writing, hold Control and press Z.

Care must be taken when using TAEd. First of all, it makes very extensive use of ANSI control sequences, and many terminals do not have all the necessary commands fully implemented. For example, certain MS-DOS terminal programs have difficulties with some of the codes sent. Most Amiga terminals, however, will work just fine with TAEd. You should note that TAEd adjusts its display depending on the screen size of the caller. Therefore, if the current caller has his/her screen length set larger than what will fit in your terminal window, TAEd's display may not look right to you. Rest assured, that provided the user's screen length is set correctly, it will look fine for him/her.

See Section B.1 [TAEd], page 103 for more details on commandline usage and online commands.

NOTE: A full-screen editor can only be used if you are a registered TransAmiga user.

3.8.3 Local Editor

TransAmiga also allows you to set the editor to use in local mode. Unlike the line editor and the full-screen editor, this editor does not have to be designed for use over the serial port. All it needs to do is to save the finished text under the appropriate name, which can be passed using %f as a part of your editor command. If you wish to quote lines when replying, the quote buffer is held in the file passed using %q. Most editors have the ability to execute macros on startup, so you may wish to create a startup macro for use with TransAmiga that inserts the quote buffer at the top of the message text.

Note that there are some problems with editors that auto- detach themselves from their parent CLI process (eg. CygnusEd Professional). TransAmiga will think that they have exited almost as soon as you run them, because they return immediately. Luckily, most editors that auto-detach like this have an option to over-ride this, so that they do not return until they have exited. You will need to consult your editor's manual on this.

3.8.4 Reply Quoting

When replying to a message TransAmiga allows anyone of the three editor types to insert quoted lines from the original message into the text. In the line editor, entering `\V` allows you to view the lines in the quote buffer, and `\Q` allows you to insert them. In the TAEd, you press `CTRL-Q`, which opens up the quote window. With the local editor, you may need to write some scripts or macros to load the quoted text into the editor.

At the beginning of the quote buffer is always the quote lead-in line as you setup in TransCfg for that message area. It is usually good practice to insert this line into your message right before you quote any actual message text, as it tells people reading the message exactly what you are about to quote.

3.9 File Transfers

3.9.1 File Listings

TransAmiga keeps a file called Files.BBS in each file directory. This file contains all necessary information about each online file to display the file listing and file information. It is an ASCII text file. Each line contains one file entry.

The first field is the name of the file. This is convenient because you can easily alphabetize your file listings using the AmigaDOS Sort command.

When the file listing is being displayed from within the BBS, you may wish to have certain comments displayed with the file listing. To do this, using a text editor, simply type in the text you want to have displayed, but have as the first character on the line be a backslash (\). You can do this anywhere within Files.BBS and it will still work. When a user goes to list the files, all lines within Files.BBS that begin with a \ will be displayed as straight text. If this text contains ANSI sequence, TransAmiga will automatically strip them for callers without ANSI.

Here's an example of what you could put in File.bbs:

```
\  ++++++++++++++++++++++++++++++++++
\  ++          AMIGA UTILITIES       ++
\  ++++++++++++++++++++++++++++++++++
\
```

```
\NOTE: All files have been compressed with LHA.
```

These lines will be displayed as-is (without the backslashes) whenever the user gets a file listing for that area. Note the lone backslash on a line for displaying a blank line.

You may wish to create a file which lists all files of your BBS that are available for download. In the TransAmiga BBS package is a program that does exactly this, called TrFL.

See Section B.8 [TrFL], page 109 for more details.

3.9.2 Long Descriptions

Users can optionally enter long descriptions of files they have uploaded. These files are stored in the file library directory under the name '<*filename*>.Desc'. For example, if the file LHA_1111.Run was uploaded, its long description would go in the file LHA_111.Run.Desc. Even if a user doesn't give the file a long description, a .Desc file will still be created. The first line of every .Desc file contains the name of the user that uploaded it (or that user's handle, if he/she has one). Every subsequent line is part of the description text. This is basically like any ASCII text file so it can be edited outside of the BBS with a text editor.

3.9.3 File_ID.DIZ

Since v1.2 TransAmiga supports File_ID.DIZ. These are small text files with a special format describing the program in detail. They are part of the archive. If File_ID.DIZ checking is enabled, TransAmiga will try to find them at upload time and put the description into Files.BBS and the long description.

3.9.4 Types of Transfers

There are 3 kinds of file transfer modes in TransAmiga: batch, local and normal. A local transfer can be done only when you are logged on locally, and there is no such thing as a local download, only local upload. Batch transfers allow you to specify several files at once to transfer. With normal transfers you must specify one transfer at a time. Local uploads are done when the SysOp would like a file he/she has placed in a file area directory to appear in the file listing. Batch transfers are used by protocols such as Zmodem (see the section on TransCfg to find out how to designate a protocol as batch). Normal transfers are for all other protocols (Xmodem, Jmodem, etc.).

To simplify local uploads, a tool called TrUFL is included.
See Section B.16 [TrUFL], page 113 for details.

3.9.5 TransAmiga XPR Support

TrXPR is an external program that TransAmiga uses for performing file transfers. It uses the XPR transfer protocol libraries, which is a standardized way for implementing transfer protocols on the Amiga using shared libraries. When you first setup TransAmiga, you were told to simply accept TransCfg's default protocol commands. All of these used TrXPR and XPR libraries.

See Section B.18 [TrXPR], page 114 for mor details.

3.9.6 File Credit

One option in TransAmiga is upload:download ratio. If file ratio is chosen , for every so many files the user downloads, he/she must upload one file and will not be allowed to download again until he/she has done so. If byte ratio is chosen, the above is calculated using the amount of bytes the user has uploaded rather than just counting the files.

This is the only limitation imposed on a user's file transfers.

3.10 TransAmiga User Maintenance

Unlike other BBS packages, TransAmiga v1.2 does not have a built in SysOp menu. Instead, it has a separate utility called TrMaint. This actually works better than having a built in SysOp menu, as it removes code from the main program that is only actually used by one or two users of a BBS, thus saving memory and cutting loading time. It is also advantageous, as TrMaint can be run not only through the BBS as a door, but also from the AmigaDOS shell, thus providing a full-featured offline user maintenance utility.

Since v1.1x of TransAmiga, a new user maintenance program for online use only is included: SMS by Yves Rausch (currently V2.02). It is recommended to use this package for online maintenance as it has more features than TrMaint and gives you more security against illegal usage of the maintenance functions.

3.10.1 Running TrMaint as a Door

To install TrMaint as a door from within the BBS, you will have to create an ARexx macro that starts it up (see Section 3.10.3 [Sample ARexx Macro], page 30). The sole purpose of this macro will be to send the appropriate command to AmigaDOS to start up TrMaint. This command is:

`TrMaint -c<`*config file*`> -n<`*line number*`>`

Where `<`*config file*`>` is the same one that you used to start TransAmiga with, and `-n` is the number identified with the current BBS line (this will always be 0 in a single line system). This last parameter must be provided, as it tells TrMaint that it will be running through the BBS and not through the CLI.

3.10.2 Running TrMaint from CLI

This is done in much the same way as described above, except that you do not pass the BBS line number. Leaving this out tells TrMaint that it should use the current CLI. All of the same functions are available regardless of whether TrMaint is run from the CLI or through the BBS, except for the fact that TrMaint can only be "multi-line friendly" when being run from the BBS, so TrMaint should only be run from CLI when there are no users currently online.

3.10.3 Sample ARexx Macro

This is an example of the type of ARexx macro you would use for starting TrMaint as a door from the BBS.

```
/* TrMaint.trans, starts TrMaint */
options results
/* Find out what line we're coming from, and what config
 * file this line is using.
 */
SYSTEMINFO 8 ; bbsline=result
SYSTEMINFO 9 ; cfgfile=result
PRINT 'Entering the SysOp Maintenance module...'
NEWLINE
/* Start up TrMaint by sending the proper command */
address command 'BBS:Bin/TrMaint -c'cfgfile' -n'bbsline
exit
```

3.11 TransAmiga Multi-Line

A multi-line BBS system is simply a BBS that has modems hooked up to multiple phone lines, so that it can handle more than one caller at the same time. This is about where the simplicity of multi-line ends, as there are numerous complexities.

What makes multi-line so complex is the fact that the BBS must some how arbitrate access to its resources, as inevitably two users on different lines will want to perform operations that conflict with each other. For example, trying to read a message that is currently being saved, and downloading a file that is still in the process of being uploaded. TransAmiga makes sure that when it wants access to something, that another copy of TransAmiga already running is not trying to access it at the same time.

3.11.1 Setting Up Multi-Line

Multi-line is setup in TransAmiga simply by creating a separate master system configuration file for each line. All other files, including other configuration files, messages, bulletins, text files, etc. can be shared by multiple lines. So you do not have to configure the message areas for each line, you do not need a separate text directory for each line, etc. Only the master configuration files need be different.

If however, you do want certain things to be different, you may. For example, perhaps you want one line of your BBS to serve MS-DOS users, and the other to serve Amiga users, you could have different file and message configurations for each line. Or perhaps, if you wanted to require a fee for access to your additional lines, you would have a different users directory for only those users with access to that line.

Usually, most of each of the master configuration files will end up being the same. Generally speaking, only the modem configuration will differ. Most likely the serial device name and/or the unit number will be different, and perhaps the baud rates and modem commands will need to be altered. Perhaps you may want your screen configurations to be a little different also.

To actually run the other BBS lines, you just start up TransAmiga several times, specifying a different master configuration file each time, and also specifying a different number using the `-n` command line parameter. For example, if you ran two lines on your BBS, you might use these two commands to start up the two lines:

```
TransAmiga -cBBS:Configs/TransAmiga_0.Cfg -n0
TransAmiga -cBBS:Configs/TransAmiga_1.Cfg -n1
```

Notice how a different configuration files was specified, and how the `-n` parameter was used to give the second invocation of TransAmiga a line number of 1. For each line you setup, you must be certain that these two things are unique.

Remember also that TransAmiga is safe to be made resident. If you are regularly keeping more than one line running, it will save you more memory by first making TransAmiga resident.

3.11.2 Running a Local-Only Line

TransAmiga has the useful ability to designate a separate line for local use only. Even though you may actually only have one line available for online callers, you can set up a second line that you can use locally, thus allowing you to log onto your board even if there is already a user online.

All that needs to be done is to have the name of your serial device blank. All other modem related fields will be ignored, and that BBS line will not be able to accept callers. The only way to logon will be locally.

3.11.3 Other programs and Multi-Line

Remember our discussion about conflicts that inevitably arise when running a multi-line system? TransAmiga has the arbitration capabilities built-in, but it cannot control what other external programs do. You have to be very careful when running multi-line, that you only use doors that are "multi-line" friendly.

For example, an online game that created a bulletin containing the scores of the various players, would have to some how ensure that another invocation of the same door isn't writing to that file at the same time. A simple solution for this is to just create a file of a different name for each line. Another possibility would be to have the program to make it's presence known, and not allow multiple copies of itself to be running at the same time. A more elegant solution would to have a locking mechanisms (using the Amiga's semaphores is a very easy way of doing this) to lock access to certain resources when they are being used.

Having doors that aren't multi-line friendly running simultaneously on multiple lines is extremely dangerous. In the worst instances, it could create corrupt files on your drive, so be careful.

4 Customizing the BBS

Now that is has been explained how to use the BBS, it is time to go into how you can customize the BBS to look and act the way you want it to.

TransAmiga allows you to do this in a number of ways. You can change the key, the access, and the text for every single command in the BBS; you can turn off commands that you don't want. You can make your own menus, with different ones for ASCII and ANSI users. And you can customize the BBS text files to give them the look you want.

4.1 Changing Commands

TransAmiga gives you the option to change any menu command key you like, enable and disable commands for all users or just for certain access levels and even add new commands.

4.1.1 The Command Files

Each command in TransAmiga can be customized. You can assign your own command key, command text, and a minimum access level to each command. TransAmiga gets information about this from special command files for each menu. You edit these command files to customize the menus. Each menu has its own command file, these all have the suffix ".Cmds". You can edit these with any ordinary text editor.

There are command files for all of TransAmiga's menus. The files are called:

```
Main.Cmds         - Main menu commands
File.Cmds         - Files menu commands
Message.Cmds      - Messages menu commands
Bulletin.Cmds     - bulletins menu commands
Settings.Cmds     - Settings menu commands
Extra.Cmds        - Extras menu commands
```

All of them should reside in your configurations directory, usually '`BBS:configs/`'.

Each file follows the same format. The first line contains the title of the menu. This is used in the default menu prompt, and the default menu command listing that users see. Next come the command entries, one for each on the menu. Each command entry takes up one line and consists of the key used for the command, the text associated with the command, the access level required for the command to be available to a user, and the flags required. TransAmiga expects the entries to be in a specific order, so do don't mix commands around. Leading spaces are stripped, and blank lines and lines beginning with a semi-colon (;) are ignored.

This is an example of what '`Main.Cmds`' might look like:

```
; The is an example command file for the Main menu.

; The first line contains the title of the menu that will
; appear in the prompt:

Main Menu

; After that comes the command entries themselves.  First
; comes the key to be pressed for the command, then the
; name of the command, followed by the access required.
; Use the asterisks as a guide to the placement of the
; fields.

; ********************** *** ********
T Time                     0 --------
B Bulletins               10 X-------
Y Yell at Tim              0 X-------
C Change Setup            10 X-------
F File Areas              10 X-------
G Good-Bye (Log off)       0 --------
H Help With Commands       0 X-------
J Join TransAmiga          0 --------
M Message Areas           10 X-------
U User List               10 X-------
L Leave Feedback to Tim    0 --------
O Online programs         20 X-------
V Version Info             0 --------
W Who's Online?            0 --------

; THE END
```

ATTENTION!

There are a few important things to note. First of all, the length of each field is fixed and must not be changed. If the spacing is altered, the BBS will probably fail when it tries to read the file.

The number of entries in each command file is fixed, with the exception of the command file for the bulletins menu ('`Bulletin.Cmds`'). The number of entries in it depends on the number of bulletins you set up. It has three basic commands then one command for each bulletin. So if you have 7 bulletins, your '`Bulletin.Cmds`' file will need 10 entries.

It is best to take the sample command files provided and edit them, to be sure you get the spacing right and the order of the commands right. Be sure to have a backup of the examples somewhere, so if you do mess up a command file, you can go back to the original.

4.1.2 Disabling Commands

It is not necessary to for all commands to be available to users. Indeed many commands are intended only for the SysOp. Some commands you may wish to disable entirely (for example, if your board is for messages only, you will want to disable the command that takes you to the file libraries). To disable a command simply give it an access level of 256, and a key that is not being used by any active command.

4.1.3 Adding Commands

You may add additional commands to any '`.Cmds`' file (see Section 4.1.1 [The Command Files], page 33). This is done by creating additional '`.Rx`' files with the corresponding menu name, e.g. '`Main.Rx`' or '`Message.Rx`'. To use this option, you actually need ARexx, as each new command simply starts a corresponding ARexx script to do the things you want it to do.

See Chapter 6 [TransAmiga and ARexx], page 57 on how to use ARexx with TransAmiga.

4.2 Customizing Text Displays

4.2.1 Text Files in TransAmiga

At numerous points during its operation, TransAmiga will use standard ASCII text files to display things. You create and edit these yourself to give your BBS a customized look. The following is a list of all text files that TransAmiga will display at one point in time or another. All

of them are optional, if TransAmiga can not find a text file, it will simply continue on with the next item of business.

TransAmiga text file have in general the following syntax:

<*filename*>[.<*access level*>][.<*language*>][.(ANSI|RIP)]

<*filename*>
: is the name of the text file TransAmiga should use. They are described below.

<*access level*>
: has to be a 3-digit number ranging from 000 to 255. This gives you the possibility to create different texts and displays for different access levels.

<*language*>
: can be used when you have setup the BBS to support different languages. *language* must have the same spelling as defined in '`language.cfg`'.

`.(ANSI|RIP)`
: will be used when the user has ANSI or RIP active.
see Section 4.2.2 [ANSI Graphics], page 38 and Section 4.2.3 [RIP Graphics], page 39 respectively.

Now the list of filenames TransAmiga recognizes automagically:

`Connected`
: As soon as a caller connects, this file is displayed. Generally, this file should just be a brief welcome message to the user, and not a fancy title screen. TransAmiga can detect if the caller is using an ANSI terminal, so an ANSI version of this file can be displayed, even though a user isn't logged on at the time, and the caller's ANSI setting is unknown.

`NoTime` If a user who actually logs on has no time left for that day he is shown this file if present in the text directory. It is displayed before the message that is taken from the language file.

`TooSlow` If a caller connects at a baud rate lower than what you have specified as your minimum rate, that caller will be shown this file and then be logged off.

`Title` This is the main title screen for the BBS, displayed after a user successfully logs on. Generally you try to make this an attractive screen telling users a little about your BBS.

`Guest` When a user logs on who isn't already a member, this file is displayed. You should use this to outline the purpose of your BBS, instructions for gaining access, and the rules that they must follow.

`News` This is the last file displayed in the logon process, right before the main menu is displayed. This is the place to put your news flash type information.

`Join` This file is shown when a user selects the command from the main menu to join the BBS. Usually this is just a brief outline of the requirements for gaining access,

`Paging` Once a user selects the page command from the main menu, they will see this file. This file could be used to say what hours the SysOp is likely to answer the page at, and also to warn off those that abuse the page command.

`Feedback` When a user issues the Feedback command from the main menu, this file will be displayed immediately before the user starts to enter the message. A good place to just thank the user for any comments he/she has about the BBS, etc.

`PreUpload`
Users see this just before starting an upload. This is a good place to remind them of the rules of uploading, and perhaps a little thank you for their contribution.

`PreDownload`
This is what users see right before downloading. This file is a good to remind users not to be file sponges, and if you have an upload:download ratio, to display that.

`Protocols`
If present, this will replace the menu that is displayed to users when selecting transfer protocols.

`SendMsg` Whenever a user goes to enter a message (either through the Write Message command, through feedback, or when replying) this text file will first be displayed. It is good for some quick instructions before writing a message (such as a reminder not to address messages to "SysOp" in echomail areas).

`ReadMode` If present, this will replace the menu that is displayed when a user goes to select the message reading mode.

`GoodBye` When a user has confirmed his/her desire to logoff, this file will be displayed. A "Thanks for calling" is usually appropriate here, and perhaps a brief listing of other boards worth calling

`PasswordFailed`
Doesn't actually belong here and has to be named exactly this way. If this file exists and the user failed to enter the password a mail using this file as body is created. The user should be told to change his password as that faild try could be a hack of someone else.

`NoTimeLeft`
: It is displayed when the user's onlinetime is exceeded for the actual day.

You can use just about any text editor (Ed, MEmacs, CEd, etc.) to make these files appear in any way you wish. For ANSI files, you will need a special ANSI editor, such as HyperANSI or LaDraw. You are encouraged to customize these as much as you wish. When displaying text files, TransAmiga will automatically convert End-of-Line to a Carriage Return & Linefeed.

Most text displays in TransAmiga can be aborted by pressing the Space Bar or holding Control and pressing C, unless this has been disabled (see Section 4.2.5 [Imbedded Control Sequences], page 39).

Because of the relative slowness of floppy drives, when a text file is being displayed, slight pauses will be noticed as the drive steps. This can be avoided by setting up your text path to be in ram:, and then in your Startup-Sequence, creating a text directory in ram:, then copying all of the BBS's text files into that directory, i.e.

```
Copy TransAmiga:Text ram:Text all
```

4.2.2 ANSI Graphics

Files ending with .ANSI contain ANSI graphics escape codes. Anytime TransAmiga tries to display a text file to a colour with ANSI, it will automatically add the .ANSI suffix and first look for that file, and fall back to the original filename if that file is not present.

ANSI is a graphics protocol often used in telecommunication for changing colours, moving the cursor, etc. These are displayed only when a user has the ANSI Graphics option on. The escape codes make these files rather cumbersome to edit in a text editor, so you will probably want to get your hands on a special ANSI editor. One that will get the job done for you is called Prism. If you have access to an IBM clone with at least CGA graphics (almost all of them have that), you can use an MS-DOS ANSI editor, which in general tend to be much more powerful than their Amiga counterparts (the Amiga system has little use for ANSI graphics, they're used almost exclusively in telecommunications).

Note that there are some differences between Amiga ANSI and IBM ANSI. IBM ANSI is more standard, and is capable of displaying 16 colours, while Amiga ANSI (handled by the console.device) only does up to 8 colours, but can display bold, underline and italics text. TransAmiga uses its own

custom ANSI routines which can handle both Amiga and IBM ANSI codes (with the exception of IBM ANSI's flashing text and keyboard remapping).

The fonts provided with TransAmiga have the upper set the same as the IBM character set - not like the Amiga's - so the IBM ANSI line drawing characters are available.

See Appendix C [Amiga ANSI Codes], page 117.

4.2.3 RIP Graphics

RIPscrip is a relative new method of displaying graphics in a BBS. The caller needs a special RIP terminal program (like ProTermRIP). Like ANSI sequences, RIP will look really weird without a proper terminal program. RIP uses the idea of ANSI to control the remote terminal via special command sequences but does much more than simply using normal characters. Icons and Bitmap graphics can be used, gadgets can be klicked and many things more. If setup correctly, RIP will give your BBS a completly new look & feel.

RIP is supported by TransAmiga since v1.2, but just in a very basic way. The Graphics will be displayed correctly on the users' side, but on the local screen you will only see the control sequences (which look quite ugly :-). Further implemetation is planned for v2.0. This is just a little help for those who want to use RIP menus and textfiles with TransAmiga to have a nice GUI for the user.

4.2.4 More?

When a text file is being displayed, and it is about to scroll off the top of the screen, TransAmiga will prompt the user with something like '`More [32%]? -Y/N/C-`'. TransAmiga is first telling the user how much of the text file he/she has viewed so far, and then prompting him/her to either continue (`Y` or `RETURN`), to stop viewing the file (`N`), or to view the file continuously with no more More? prompts (`C`).

4.2.5 Imbedded Control Sequences

TransAmiga allows more than just straight text in text files. It supports a number of special control codes, that are interpreted by TransAmiga, and tell it to output certain information. These control sequences allow considerable flexibility. You can use them to cause a '`Press RETURN to`

`continue.`' prompt to come up, to display the name of the online, to pause for a few seconds, and a host of other things.

The control sequences can appear anywhere in a text file, and the file can contain as many of them as you wish. For a complete List of the sequences, see Appendix D [Control Sequences], page 119.

4.3 Menus and More

In this section you will learn how to change the look of your BBS. TransAmiga gives you the possibility to change nearly everything the BBS displays to the caller and so giving it your own personal touch. As TransAmiga also offers the usage of different languages for standard outputs, all the text displays like menus, prompts and help files should be translated too.

4.3.1 Menu Files

You actually setup up the BBS's menus when you created your command files. When the user presses ? to see a menu, the BBS constructs a list of commands for which the user has access to on the screen. This is quite useful as it displays only the commands the user has sufficient access to use.

You may however wish to design your own menus. This is all a part of giving your BBS that "personal touch". To do this, you will need to create text files for each menu that you want to be customized, and place those files in your text directory. TransAmiga allows considerable flexibility when creating you own menus. You can create separate menus for specific access levels, or a default menu for all access levels. This works pretty much the same way like creating the Text Files.

See Section 4.2.1 [Text Files in TransAmiga], page 35 for complete syntax description.

Now for the actual names of the menu files. Naturally, each menu within the BBS is going to need its own menu file:

```
MainMenu        - Main Menu
MsgMenu         - Message Conferences Menu
BlltnMenu       - Bulletins Menu
ExtraMenu       - Extras Menu
FileMenu        - File Libraries Menu
SetMenu         - Settings Menu
MsgConfs        - Menu Listing Available Message Conferences
FileLibs        - Menu Listing Available File Libraries
MenuHeader      - Displayed before the above menu files
```

Your finished menu files go into the text directory. Example menu names are: '`MainMenu.000`', '`FileMenu.255.ANSI`', '`BlltnMenu`', '`ExtraMenu.ANSI`', '`SetMenu.025.ANSI`', etc.

All the imbedded control sequences (see Section 4.2.5 [Imbedded Control Sequences], page 39) available in other text files are available in the menu files, as well as the prompt and help files.

ATTENTION! If you do not use a menu file for one menu but have a Menuheader file the header is displayed and then TransAmigas internal menu generation! This may cause funny displayeffects if the user has clearing codes active... :-) Anyway it would not look very well so I suggest to make menufiles for all internal menus if you use this feature.

4.3.2 Prompt Files

Prompt files are much like the Menu files described above, except that instead of appearing whenever the user presses ? at a menu, the appear right away (they are menu prompts). Also unlike menu files, you can not have specific prompt files for an access level, users see the same prompt file regardless of access. The file names of the prompt files are as follows:

```
MainPrompt        - Main Menu
MsgPrompt         - Message Conferences
FilePrompt        - File Libraries
BlltnPrompt       - Bulletins Menu
ExtrasPrompt      - Extras Menu
ReadPrompt        - Read Messages Sub-menu
ListPrompt        - List Files Sub-menu
ContinuePrompt    - Press Continue Prompt
SetPrompt         - Settings Menu
```

If a prompt file isn't available then TransAmiga will display a default menu showing the name of the menu and the amount of online time the user has remaining.

All Prompt files should be in your Text directory. Example file names are: '`MainPrompt`', '`MsgPrompt.ANSI`', '`ReadPrompt.ANSI`', '`ReadPrompt`', etc.

The `CTRL-K 5` imbedded control sequence is particularly useful in the prompt files, as it gives the user a quick listing of what commands he/she has available.

4.3.3 Help Files

TransAmiga has provisions for providing online help to users of your BBS. Each menu has a Help command, and when selected by a user, TransAmiga will display a help text file for that menu. These files all end in Help and all reside in your text directory. The filenames for the help files are:

```
MainHelp          - Main Menu
MsgHelp           - Message Conferences
FileHelp          - File Libraries
ExtraHelp         - Bulletins Menu
BlltnHelp         - Extras Menu
SetHelp           - Settings Menu
SysOpHelp (actually used by TrMaint, not TransAmiga)
TransEdHelp       - TransEd Menu
```

As you can see there is a help file for each menu. Although you don't actually have to have these files around, it is highly recommended that you make them. For novice BBS users, getting around can be a little difficult, so you should try to help them out with information in the help files.

4.3.4 Language Files

Almost all of the text strings displayed by TransAmiga can be found in files in the Configs directory called language files. Language files can be identified by the .Lang file extension. Language files are actually just simple text files containing the various prompts and messages that the user will see. Things such as the 'Enter your full name:' and 'Privileged message?' prompts are in this file. By editing the default language file, you can change the text that the user sees.
If several .Lang files exist in your config directory, the user can choose on of the languages, either

in the first callers procedures or in the setting menu.
If you have ARexx, you can use the LOADLANGUAGE ARexx command to switch between different language files.

Each line of a language file contains one text string. The only way you can tell which prompt appears when in the BBS is from the text itself and from experience with using the BBS. When editing it, you must be extremely careful not to remove lines, or to shuffle lines around. And most importantly, be sure you have a copy of the original Default.Lang file backed up somewhere, so that you can revert to that should something go wrong.

Editting language files is for advanced users only, do not alter language files unless you know what you are doing.

Since v1.2 it is possible to have .ANSI or .RIP versions of the language files present.

4.4 Access Levels

Access levels are very important for the security of a BBS. For example you can deny guests the option to upload programs to prevent introduction of virii. Or you could restrict write access to some message areas to prevent newbies making trouble in a net like Fido. There are so many possible uses of access levels that they cannot be described all. You will find them out by yourself as you get more convenient with TransAmiga and BBSing.

4.4.1 User Access

TransAmiga has 256 different access levels numbered from 0 to 255. Access levels are useful for a variety of purposes. For example, allowing assistant SysOps access to BBS maintenance functions, restricting new non-validated users, etc.

4.4.2 Access Flags

Access flags are a second level of access control in TransAmiga. The BBS is designed such that the access level should be used as the primary means for controlling access, but access flags give an additional level of control.

Access flags are a set of eight "flags". For a user to have access to, for example, a certain file area, he/she must have at least the same flags set as are required for that file area. For example, say the flags for that area are:

```
X--XX----
```

(X's indicate the flag is set, -'s indicate it is unset)

Now, a user with flags:

```
X---X-X---
```

would not have access, because not all the necessary flags are set; the fact that he has another extra flag set is of no consequence in this situation. However, a user with flags:

```
X--XX--X-
```

would have access, since all the necessary flags are set.

4.4.3 New Users

When a user logs on for the very first time, he/she is given access 0 which denotes a user who has just logged on. This user is known as a "Guest", one who has just logged on and is looking around. Therefore, generally speaking, the functions that a guest should have access too should be quite limited. You should never manually set a user's access to 0, as this will be considered by TransAmiga to be a deleted user. 1 is the lowest access level.

If the guest wishes to become a member of the BBS, he/she must select the command to join the BBS. Upon completion of this questionnaire, the user receives the new user access level, time limit, and upload download ratio. How much access is given to this kind of user depends on your own preferences. Remember it is still that person's first call and you have not had a chance to validate him/her. Some people like to run an open system and give first time callers full validated access, while others prefer to keep them restricted. You can set these values with TransCfg (see Section 2.4.1 [General], page 10).

4.4.4 Privileged Access

Privileged access is anything equal to or above what you defined as privileged access in TransCfg.

You should be very careful whom you give privileged access to. Generally speaking, in addition to yourself there should only be one or two more people with privileged access, usually your CoSysOps, if you have any.

4.4.5 A Note on Users Rights

Human rights granted to people differs in different countries. In most countries though, everyone must be granted access to your BBS, even if you wish to run a private system. You may not discriminate against a user because of race, sex, age, etc. nor may you limit any of their other rights. The law still applies in the BBS world!

5 Fidonetting

TransAmiga supports FidoNet as a point or a node. The difference between a node and a point is basically that a node is more independent. Not much time will be spent on using TransAmiga as a point, as it is not all that much difference, and the advantages of being a node are such that it is recommended that you become one.

Please consult the FidoNet.man by the programmers of TrapDoor for details and background information about FidoNet. This manual can be found on many Amiga Fido BBS' and of course in the TrapDoor distribution package.

5.1 What you need

To actually have a complete running FidoNet setup, you'll need more than just TransAmiga, as it is only one component in a total FidoNet setup. Unlike other BBS software packages, TransAmiga does not attempt to "re-invent the wheel" by attempting to write equivalents of perfectly good FidoNet tools that already exist. History has shown that this leads to many compromises in both features and performance.

So in addition to the BBS, a basic FidoNet setup with TransAmiga involves the following software:

1. The mailer (sometimes called a session handler or a frontend)
2. The mail processor (also called a mail packer, or mail tosser)
3. The message reader and editor.

(Most complete setups will include more software. If you are feeding echomail conferences to points or other nodes, an AreaFix utility is needed. If you are handling file echoes, a Tick utility is needed. These kinds of setups are much more involved, and tend to be highly personalized, so it is beyond the scope of this document. We will just go into a basic setup.)

As it is absolutely recommended that you have some experience either as a point or a node before setting up TransAmiga for FidoNet, basic things are not covered in this manual. Please refer to the manuals of the mailer and mail processor for these things.

At the moment, there are two major mailers used by Amiga Fidonet Sysops: TrapDoor by Maximilian Hantsch and Martin Laubach and GMS by Mirko Viviani. Both are very reliable.

TrapDoor was the No 1 for a long time but in the last few months, GMS got very popular as it's developement is very fast and some nice new features are implemented.

There is another mailer in developement, JamMail by James McOrmond. It looks quite interesting but still needs developement. It may be enough for your needs and over all, it's free. Be warned that you will need to register TrapDoor before you can use it properly in a node environment. Be sure that the version of JamMail is newer than 31/10/94, and TrapDoor is at least 1.85, GMS 1.1.

As far as mail processors go, you have a few more choices. There are several major ones, including: MailManager, Foozle and TrapToss though Foozle is recommended but quite outdated now. MailManager is state of the art for a node setup.

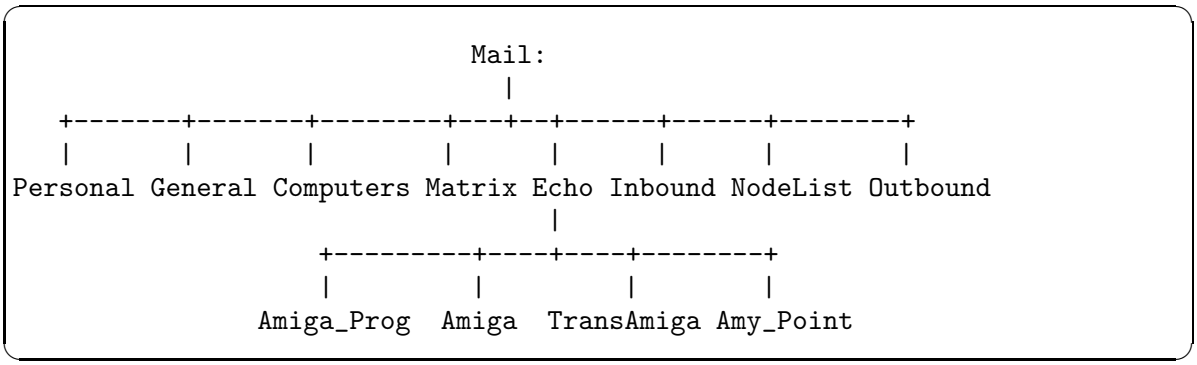
5.2 Setting up FidoNet

In this part is it assumed that you already have FidoNet access and that you are running a Point or a Node.

One of the first things you do is set up your FidoNet addresses. In TransAmiga you may have several addresses (for example, if you belong to more than one network, or if you hold a position such as Net Co-ordinator), though most people will only have one address. Network addresses in TransAmiga are 5-dimension, consisting not only of the normal zone, net, node and point, but also of an optional domain, which indicates the name of the network. Usually this is not needed unless you are in several networks. You can also set a pointnet (also known as fakenet) to go with each address. Use this only if you are a point with a pointnet, nodes that have points should not use this. Also, if you are a point and your bossnode is running software that does not require a pointnet, you may enter a pointnet of zero, and TransAmiga will create the proper 3-dimensional PATH: kludge lines.

The rest should be straightforward enough, you simply enter your address, then your inbound, outbound, and nodelist directories. Then all you have to do is setup the FidoNet areas. This is just like adding any other kind of message area, except you will need to say which one of your addresses to use (normally just number 1 if you have only setup one address), and give an origin line for echomail areas. When doing so, do not enter the '* Origin:' string, or your address at the end, as this will be added automatically. If you've installed '`Foozle`' as mail processor, you can use the same message base (i.e. directory structure and #?.MSG type message files). If you've installed '`MailManager`', you have to use Plain or Indexed as a MsgAreaType within '`MailManager`' to use the same message base within TransAmiga.

Your mail directory tree may now look something like this:



5.3 Events

The command used to start up the BBS requires some more indepth explanation. In TrapDoor, this is set using the BBSCOMMAND.

See Appendix A [Examples], page 97 for an example scripts to start TransAmiga from TrapDoor.

At first, you may think that all that has to be done is to simply start the BBS in immediate logon mode at the baud rate of the call that eg TrapDoor answers. But this is too rigid for our purposes, and does not accomplish all we need to.

To run a fully-operational node, certain other things have to happen, called "events". For example, at some point in time you will have to call out to receive your mail bundles. Then you'll need to unpack those bundles, and you'll also have to create bundles of your own to send out.

We will divide up the events into two categories: time driven events, and BBS driven events. Time driven events are things we want to happen at a certain time in the day. For instance, you usually want to have your system call out for mail during the night sometime. BBS driven events are things that need to be done due to something a user on the BBS has done, for example entering a netmail message, which would need to be packed up and sent off.

Lets get a basic listing of actions that initiate the events, and how we want things to react when these events happen:

```
Action:                        Reaction:
- User enters a FidoNet        - Tell mail processor to
  message                        export
- Mail is delivered            - Tell mail processor to import
- Zone Mail Hour               - Tell mailer not  to  allow BBS
                                  callers or file requests
- Time to poll for mail        - Tell mail  processor to export
                                  any remaining  messages.  Tell
                                  the  mailer  to call all nodes
                                  with  waiting  mail,  and also
                                  the   feed   for   our   mail.
                                  Import   any   received   mail
                                  bundles.     Renumber  message
                                  areas.
```

This is just the basic set of events, once you get things running you will probably want more.

5.4 Exporting Mail

The most simple event is exporting mail when a user has entered a FidoNet message. To do this, we have to setup the `Logoff.trans` properly by using the `NETMAIL` and `ECHOMAIL` ARexx commands to detect if the user entered a mail that needs to be exported. You can find such an ARexx script in the '`rexx/`' directory of the distribution package.

5.5 Importing Mail

If you are using TrapDoor, it has the ability to execute a command after every mail session, so it would be easiest to have your importing done there. Just have your AFTERSESSION command execute a script that call the mail processor to import mail.

See Appendix A [Examples], page 97 for an example ARexx script.

5.6 Time Driven Events

Now we must handle the other kind of event, one caused by a certain time of the day. To do this, we must first have a means of finding out when that time of day is. A type of program called a

"Cron" executes commands a certain time of day, which may be easiest for you. There are several of these utilities available. Two of the more popular ones are CyberCron and TPTCron.

All you have to do is set these to start up scripts to handle your various events at the correct time of day.

5.7 Other Mailers

TrapDoor, GMS and JamMail are the only mailers currently available that are capable of supporting a BBS in a full-node environment. This may change however, and hopefully the design of TransAmiga is such that getting any new mailers to work with it won't cause a problem.

5.8 TransAmiga Update Message Pointers

TrUMP is a special utility for TransAmiga, it stands for TransAmiga Update Message Pointers. What it does is to save the high and low message numbers in a FidoNet message directory, so that TransAmiga does not have to perform a time consuming message base scan when users are online. TrUMP is intended only for FidoNet areas, as they will often be accessed by programs other than the BBS, but it is also safe to use on local areas should they need to be rescanned for some reason.

TrUMP needs to be run any time that the low and/or high message number in an area changes. It must also be run after the initial setup of any matrix or echo conference.

Run TrUMP right after: importing, renumbering, deleting, and after messages have been added by some program other than TransAmiga.

If you do not run TrUMP on affected areas, TransAmiga will have incorrect values for the high and low message numbers, which could cause messages to be overwritten.

TrUMP must be run from the CLI or shell as follows:

`TrUMP <`*mail dir 1*`> [<`*mail dir n*`>...]`

All you do is give TrUMP a listing of the directories to be rescanned. Foozle users should note that there is an alternate usage form for TrUMP, see later on for more details.

If you run TrUMP while TransAmiga is running, TransAmiga will not immediately become aware of the new message pointers (unless you use the RESCAN ARexx command, described later), however, TransAmiga will re-read the message pointer file before it adds any messages. Therefore it is perfectly safe to run TrUMP while TransAmiga is running (but see the important section below on TrAL).

5.9 TransAmiga Access Lock

As you become more advanced with your system, you may find times when it would be convenient to make changes to the message bases while the BBS is still running, and possibly, users are online. Usually, these sorts of practices are totally out-of-the-question because of the threat of your program trying to access a file at the exact same time as the BBS does. For example, when running FidoNet, it is usually necessary to disallow humans from logging onto the BBS while the mail processor was running.

TransAmiga offers a solution to this problem in TrAL, which allows you to temporarily tell TransAmiga to lock out writing to the message bases so that other programs can access them safely even if there are users online.

With the message areas locked, other programs can then work with the message areas, doing things like importing messages, etc. It is not, however, safe to run any process that modifies existing messages when the message areas are locked, only things that add new messages.

To use TrAL, you start it from the shell in the following manner:

```
TrAL [L|U]
```

Tral L tells TransAmiga that users are Locked from writing, and TrAL U Unlocks them. Be sure to match every TrAL L with a TrAL U, or users won't be able to write messages.

Typically, scripts that call the mail processor should first do a TrAL L, go about their business, run TrUMP if necessary, then run TrAL U.

5.10 Renumbering

Sooner or later, FidoNet areas will have to be renumbered. Many people like to do this on a regular basis, others wish to do this only once in a while.

In order for individual users' last read pointers to remain accurate after a renumber, the tool you use for renumbering must update TransAmiga last read pointers, which are contained in files called LastRead.BBS in each message directory. Check the documentation of your mail processor to see whether or not it supports updating of TransAmiga's last read pointers. If it doesn't, tell the author he/she should fix that. For the time being you'll have to either use another program to handle renumbering, or simply don't renumber at all.

Although Foozle does not directly update TransAmiga last read pointers, TrUMP may be used in a special manner so that last read pointers are properly updated. To do this, you must use Foozle's FzI utility to edit your External commands. Set the after-renumber command to something like:

```
BBS:Bin/TrUMP %r %d
```

What you need to do is have Foozle execute TrUMP immediately after an area has been renumbered, passing it Foozle's special renumber area and the directory that has been renumbered. It is absolutely essential that you have Foozle setup in this manner, else lastread pointers will not be accurate after a renumber.

NOTE 1: Every time you renumber, TransAmiga must rescan the message areas or it will have incorrect values for the high and low message number. For this reason, you should never renumber when users are online, even if the message areas have been locked with TrAL, wait until after a user has logged off and ensure that no one calls while the renumbering is taking place. If you exit then re-run TransAmiga after every call, the message areas will be scanned when it starts up; if you have TransAmiga waiting on its ARexx port, then you must send it the RESCAN command. Remember also to run TrUMP right after the renumbering

NOTE 2: This only applies to actual renumbering of message, not just the deletion of older messages.

5.11 Offline Message Readers

The message format used by TransAmiga conforms exactly to what is outlined in FTS-0001. Therefore, any message reader that uses this message format should be usable in a TransAmiga

FidoNet environment, such as Chameleon or PointManager (the package Spot uses its own message format which is completely incompatible with TransAmiga and is just a point package).

TransAmiga also has special support for the Foozle Mail Manipulation System by Peer Hasselmeyer. Foozle is a super fast shareware integrated FidoNet message reader and mail processor. Foozle supports a custom message format that is a variant of the FTS-0001 format. TransAmiga will automatically recognize a Foozle message, and read in the extra header information that Foozle messages contain. TransAmiga still writes messages in the standard FTS-0001 format, so Foozle users will have to use its RESCAN command before it does anything else with the message areas, since Foozle will not automatically recognize FTS-0001 messages. Also, Foozle does not update the TransAmiga last read pointers, so the TrUMP utility described above must do this. This does not end up being any different than with other mail processors, as TrUMP has to be run anyway after a renumber.

5.12 Using FidoNet

Using FidoNet simply involves reading and writing messages in FidoNet areas. This is basically the same as in local and all-privileged areas, except that extra FidoNet specific information is inserted into the messages.

When entering messages in a Matrix area, you will first be asked for the FidoNet address you wish to send the message to. When you enter a valid address, TransAmiga will automatically look up this address in the nodelist, and when entering the name of who you wish to send the message to, the default name will be the SysOp of that node (of course you can edit this).

Netmail is by default sent routed. TransAmiga leaves the topic of netmail routing up to your mail processor (Foozle has routing capabilities built in), however privileged users are given additional control. If you're privileged you will also be asked if you would like to send the message "Routed", "Crash Mail", "Hold" or to send a "File Attach". Most mail processors will look at their routing information to determine what happens to mail marked as routed, while crash mail will always be sent directly to its destination, and mail put on hold will not be delivered, but rather kept until the destination node contacts your system. File attaches sends a file of your choice along with the message and are always marked crash.

Before a user can actually send the message, TransAmiga will check the cost of sending the message against the users netmail credit. The user will not be able to send a message if he/she does not have adequate credit. User gets back netmail credit if he chooses 'routed' or 'hold'. As

there is no incomplicated way to prevent substraction of the credit before the message is entered the msgcosts are returned later...

Remember, in order to keep track of costs correctly, the message cost for nodes that you route mail to should be set to zero in your nodelist processor configuration.

6 TransAmiga and ARexx

Arexx is a very powerful macro language available for the Amiga. If you have OS2.x or above, then you already have ARexx, otherwise you will need to purchase ARexx.

If you do not have ARexx, then you are missing out on some of TransAmiga's most powerful features. Nonetheless, TransAmiga does make attempts to compensate for this and is still fully functional. However, it is highly recommended that you upgrade to at least OS2.04, which comes with ARexx.

No lessons will be given in programming in ARexx, but for those not familiar with it, ARexx is an interpreted language that includes the ability to send commands to any other running program that has opened a suitable ARexx port. With a little imagination, tremendous things can be done through ARexx.

6.1 Using ARexx with TransAmiga

It is TransAmiga's extensive ARexx interface that makes the BBS so powerful. It allows you to easily program in new functions in addition to the built in ones. You can change the way existing commands work, add new commands, and basically anything you want. It makes TransAmiga a completely programmable BBS.

You will probably find that after setting up the BBS, ARexx will soon start to play a very important role in the running of your BBS. The ability to customize, and the ease with which you can customize, are so great that you will probably find yourself relying on ARexx to make the BBS act the way you want it.

At first you may feel that having to use ARexx like this is too much work, and that the BBS should be able to handle more things without ARexx. However, in order to maximize the SysOp's ability to make his/her own decisions about the way his/her board appears, ARexx must be used, otherwise you would always be running into limitations of the BBS software. After writing a few ARexx macros yourself, once you get the hang of it, you will see the enormous potential the TransAmiga's ARexx interface offers.

When TransAmiga starts up, it opens an ARexx port called 'TRANSAMIGA.<*line number*>' where it can receive ARexx commands. TransAmiga sports over 100 ARexx commands that allow you to access virtually every single function of the BBS.

By convention, all TransAmiga ARexx scripts should end with the '.trans' suffix so that they can easily be recognized as ARexx scripts for use with TransAmiga. All TransAmiga ARexx scripts must reside in your BBS ARexx directory.

There are two ways to get TransAmiga to run ARexx macros. One way is as the cause of a user action. For example, after logging on, the ARexx macro 'Logon.trans' is run. The second way is by installing an ARexx macro as a command on a menu in TransAmiga using the special .Rx files in the TransAmiga configurations directory.

The following ARexx macros are started because of user actions:

`Banned.trans`
: after the line from the .lang file is displayed (banned user tried to log on) and before logoff.

`CChat.trans`
: replaces standard chat mode. Internal chat started when EXIT 10 is used.

`Connected.trans`
: executed as soon as a connection is established

`Editor.trans`
: immediately after the user exits the editor. The text entered can be found in t:Editor.Temp (for line 0) or t:Editor_<line number>.Temp (for line numbers other than 0).

`File.trans`
: when a user tries to go to the file area. If returned 10 or higher, it will return to the main menu.

`GoodBye.trans`
: after a user selects the good-bye command. Returns to the BBS when the script uses EXIT 10 or above.

`Join.trans`
: after the user has answered the questions for the Join command. If returns failure, TransAmiga will not raise the user access.

`LogOff.trans`
: after the user has actually been logged off.

`Logon.trans`
: after displaying the Title text file.

`Message.trans`
: when a user tries to go to the message area. If returned 10 or higher, it will return to the main menu.

`PassFail.trans`
: after the PasswordFailed mail generation and before logoff.

`Upload.trans`
`Download.trans`
: executed immediately after an upload or download repectively.

None of these are required, and TransAmiga will first check to see if they are present before trying to execute them. If they are not present, TransAmiga has some default actions that it will take. If the Logon.trans ARexx macro is not there, then the user will shown a list of new message posted since his/her last call, and also be prompted to search for new messages sent to him/her. If the File.trans or Message.trans macros are not found, then the user will be shown the list of available areas, and be prompted to select one. If the GoodBye.trans macro is not present, then the user will be asked if he/she wishes to change the one-line comment.

To add an ARexx macro as a command from a menu, simply create an .Rx file for that menu. The format for these .Rx files is exactly the same as the .Cmds files use to set up commands, described earlier, and the filenames are the same too, except that they end in .Rx rather than .Cmds. It has the key used for the command, followed by the text string to be associated with the command, followed the access level required, followed by the flags. What makes the .Rx files different is one additional field that comes after the access level with a space inbetween. Here you put the name of the ARexx macro to be executed when a user selects this command from the menu. For example:

```
; Lines beginning with semi-colons are comments.
; Blank lines are ignored, all other lines are assumed
; to be ARexx command entries.

; ********************** *** ******** ******************
P Poker                   10 X------- Poker.trans
W Wheel of Fortune        10 X------- Wheel.trans
Y Add to BBS Listing      10 X------- BBS_Lister.trans

; THE END
```

Be sure that the ARexx macro you specify is indeed a valid ARexx macro in your BBS ARexx directory, otherwise the BBS could be hung waiting for ARexx messages coming from a macro that was never run.

Since v1.11 TransAmiga "listens" to his ARexx port at all the time, while older versions only processed ARexx commands when running a macro. This means that as long as a user is online, ARexx commands will be immediately processed.

6.2 Writing ARexx Macros

Every single ARexx macro you write for TransAmiga must have the line `options results`, otherwise the macros will not function properly. This tells TransAmiga that it is OK for it to send return codes back to ARexx. As TransAmiga needs to set return codes for many of its commands, this line must exist in each and every TransAmiga ARexx macro.

Secondly, you must be able to handle unexpected user logoffs, due to either dropping carrier or an idle timeout. There are two ways to handle this: when a command returns something in the result variable, such as when querying the user for a string, TransAmiga will update the ARexx variable CARRIER. If the user is still online, it will hold the value of 1, otherwise, it will equal 0. All properly written ARexx macro must check this regularly and respond appropriately. TransAmiga also returns `###PANIC` on any command if the carrier is lost. Make your programs exiting when they get `###PANIC` returned from any ARexx command. You don't have to check all returns, but those you expect something to be returned.

TransAmiga ARexx macros need no 'address' line in them, as the port is set to TransAmiga's when the macro is started.

ARexx macros started by TransAmiga do not need to, and should not, address TransAmiga's ARexx port, as TransAmiga sets this before the macro is started. However, if you start up an ARexx macro from another source (eg. the shell) that is to talk to TransAmiga, you will first need to address the name of TransAmiga's ARexx port. TransAmiga opens a different ARexx port for each line, called 'TransAmiga.#' where # is the line number, eg. line 1's ARexx port will be called 'TRANSAMIGA.1'. Since ARexx converts all unquoted strings to upper-case, you must enclose the port name in quotation marks.

Note that ARexx commands never should use more than 999 characters at a line.

From then on, all you have to know are the special ARexx commands that TransAmiga understands. As you will see, there is a very large variety.

6.3 TransAmiga v1.0x Doors

The original versions of TransAmiga had only the most basic of door support, where a door was simply a DOS program that was run, and no interaction between the door and TransAmiga could take place. Nonetheless, some doors were written, most notably PTA by James Stewart for running Paragon BBS doors.

Doors in TransAmiga v1.0x were added as command to menus using the .Exts files, but they are now gone. Now, you must have an ARexx macro installed as a command which inturn starts the v1.0x door using the RUNDOOR ARexx command. TransAmiga v1.1 creates the old-style t:Door.Temp file for complete compatibility (new doors should no longer use the t:Door.Temp file, but should instead get information through TransAmiga's ARexx port). Note also that the name of this file depends on which line starts the door.

Here is a sample ARexx macro for starting a door via PTA:

```
/* GlobalWar.trans, starts GlobalWar through PTA */
options results
PRINT 'Loading Global War...'
RUNDOOR 'BBS:PTA -s BBS:Extras/GlobalWar/GlobalWar'
exit
```

6.4 ARexx Macro Ideas

Because of the vastness of the commands available in TransAmiga through ARexx, you are virtually unlimited in what you can do.

One of the most popular applications is online games, as these are quite popular with many BBS callers. TransAmiga's ARexx interface is ideally suited to easily writing games, but there is far more that you can do. Here are some ideas you may wish to think about:

- A callback verification program for auto-validating new users.
- Personal file space for users, so that two users can exchange files privately.
- Division of file and message areas into sub-groups (sometimes called SIGs).
- A polling booths for users to create and vote on poll topics.

6.5 Command Reference

This sections lists all the ARexx commands that TransAmiga understands. Many of them will return values of some sort. You ARexx macro can look at these values by evaluating the variable RESULT.

TransAmiga's ARexx interface is divided into two sets of commands: The first set, used most often, is made up of the commands used when TransAmiga starts an ARexx macro when a user is online. The second set, which are only used for special situations such as running a FidoNet node, are ones that are only understood when TransAmiga has been started with the -r option and a user is not currently online.

The online commands can be further divided into TransAmiga specific commands, and the ABBEREXX commands and variables. The ABBEREXX commands are actually a standardized set of commands used by several different BBS packages, allowing ARexx macros written for different systems to be run by TransAmiga. If you want your macros to be run by SysOps using other BBS packages, then you should restrict yourself to the ABBEREXX command and variable set. However, for most applications, you may find it desirable to make use of the multitude of additional TransAmiga specific commands that are available, with the consequence being that only TransAmiga systems will be able handle these macros.

6.5.1 ABBEREXX Commands

These are the commands that TransAmiga supports as a part of the Amiga Bulletin Board External Rexx (ABBEREXX) interface command set. These commands should be used when portability to other ARexx supporting BBS's is desired.

`BBSIDENTIFY ABBEREXX BBS|EMULATION|NAME|SYSOP|TERM|USER`

returns certain information, depending on the <what> parameter. Possible values are:

`ABBEREXX` which returns the current version of the ABBEREXX implementation (at the moment, '1.0').

`BBS` which returns the current version of TransAmiga is being used.

`EMULATION`
indicates what type of terminal mode the caller has selected. It returns either 'ASCII' or 'ANSI'.

`NAME` returns the name you have set for the BBS.

`SYSOP` returns the name of the SysOp of the BBS.

`TERM` which returns a string of the format <*baud*> <*rows*> <*columns*> <*line number*>, eg. 2400 80 24 0.

`USER` which returns a string of the format "<*username*>" "<*calling from*>" <*access*>, where access is one of:

- `GUEST` a first time caller, access 0
- `MEMBER` anyone with access greater than 0 and less than the privileged access level
- `CO-SYSOP` anyone with access equal to or greater than privileged access, but less than 255
- `SYSOP` access 255

An example return string could be:

```
"Sami Radwan" "Karlsruhe" SYSOP
```

```
BBSIDENTIFY ABBEREXX
BBSIDENTIFY BBS
BBSIDENTIFY EMULATION
BBSIDENTIFY NAME
BBSIDENTIFY SYSOP
BBSIDENTIFY TERM
BBSIDENTIFY USER
```

`CHECKABORT`

Returns 1 if the user has pressed one of the abort keys (`CTRL-C` or the `SPACEBAR`), otherwise 0. Takes no parameters.

`CLS` Sends the screen clearing code. Takes no parameters and returns nothing.

`GETCHAR` Waits for a single character from the user. It returns as soon as it gets something, and does not wait for the user to press `RETURN`. It takes no arguments, and the character received will be returned, or `###PANIC` if the user was unexpectedly logged off due to carrier loss, timeout, etc. Returns only uppercase characters.

`HANGUP` Hangs the online caller up, and after the ARexx macro terminates, the logoff procedure will be initiated. Note, that if it is a local logon, nothing actually happens until the ARexx macro terminates. Takes no parameters and returns nothing.

`IREADY` If there is any input waiting in the serial buffer, this command returns 1, else 0. No input is retrieved though. It takes no parameters.

`MAYGETCHAR`

Checks if they user has entered a character, and returns immediately, with the result being either the key the user pressed or `NOCHAR` if a key was not pressed. Takes no parameters.

```
MAYGETCHAR
  cmd=RESULT
  if cmd~='NOCHAR' then call do_cmd
```

`NEWLINE` Sends the newline code (a carriage return followed by a line feed). Takes no parameters and returns nothing.

`PRINT` <*string*>
Sends the given string with a newline appended. It returns nothing.

`PROMPT` <*length*> `NORMAL|HIDE|YESNO|NOYES "`<*prompt*>`"`
Prompts the user to enter text with the given length as the maximum. Three types of text input are currently supported:

`NORMAL` Normal line oriented input

`HIDE` As above, but hid user input behind dashes (-). Useful for passwords, etc.

`YESNO` A Yes or No response, with Yes as the default.

`NOYES` A Yes or No response, with No as the default.

The prompt text must be enclosed in quotes, an din order for the BBS to "see" these quotes, they themselves must be quoted (see examples).

This command returns the string entered in the cases of `NORMAL` and `HIDE` types, and either 'Y' or 'N' in the cases the `YESNO` and `NOYES` types.

```
PROMPT 30 'NORMAL' '"Enter your favourite colour: "'
PROMPT 10 'HIDE' '"Where were YOU last night?'"
PROMPT 1 'YESNO' '"Was it good for you?'"
```

`SEND` <*string*>
Sends the given string exactly as is to the user, with no translation or screening done on the part of TransAmiga. It returns nothing.

`SENDFILE` <*filename*>
Sends the given text file to the online user, the full pathname must be given. More? prompts will automatically be displayed as usual, and all TransAmiga imbedded control codes are available as usual. If the current user has ANSI turned on, it will first look for '<*filename*>`.ANSI`', and fall back to the original filename if that is not found. It returns nothing.

```
SENDFILE 'BBS:Text/Title'
```

`SETNODELOCATION` <*where*>
This allows you to change the string that indicates what a user is doing that is displayed with the List Online Users command, or the TrBLAD (see Section B.5 [TrBLAD], page 107) program.

```
SETNODELOCATION 'Playing "Kill the Capitalists"'
```

`SYSOPLOG <string>`

Creates a log entry with the given string. The date and time will automatically be added infront. It will be appended to TransAmiga's log file. It returns nothing.

```
SYSOPLOG 'Lost all his/her money at poker'
```

6.5.2 ABBEREXX Variables

Part of the ABBEREXX standard is the maintenance of three ARexx variables. These are: `CARRIER`, `TIMEONLINE` and `TIMELEFT`. `CARRIER`, as explained above, indicates whether or not there is a user online line, it contains the value 1 if a user is online, 0 otherwise. `TIMEONLINE` and `TIMELEFT` indicate, in minute, how much time the user has spent online today and how much time he/she has left for today respectively.

```
PRINT 'You've used' TIMEONLINE 'minutes today'
QUERY 'How do you feel about that?'
answer=RESULT

if CARRIER=0 then do
     SYSOPLOG 'Carrier Drop!'
     exit 10
end
if timeleft<=0 then do
     SYSOPLOG 'No more time!'
     exit 10
end
```

6.5.3 ARexx Commands

These are the commands that TransAmiga supports in addition to the ABBEREXX commands while there is a user online.

`ADDDLFILES <value>`

Adds *value* to user's downloaded files value.

`ADDDLKB <value>`

Adds *value* to user's downloaded kbytes value.

`ADDMSGS <value>`

Adds *value* to user's written messages. Is thought for scripts storing messages in external message bases to support e.g. UMS or EMS.

ADDULFILES <*value*>
: Adds *value* to user's uploaded files value.

ADDULKB <*value*>
: Adds *value* to user's uploaded kbytes value.

BROWSEFILES <*mode*>
: Allows you to select either browse or regular mode for listing files. A mode value of 0 indicates normal mode while a value of 1 switches to browse mode. Returns nothing.

BROWSEMODE <0|1>
: See LISTMODE.

CHARSTACK
: This checks if there is anything in the current character stack, i.e. whether or not the user has "stacked" one or more commands. Returns 1 if there is, otherwise 0. It takes no parameters.

```
CHARSTACK
if RESULT=0 then SENDMENU 'BBS:Text/MyMenu'
```

CHECKMSG
: Does the same as CHECKMSGS but only in the actual conference.

CHECKMSGS
: Does a search of all new messages in all available areas for messages addressed to the user or to the user's handle (if he/she has one). Found messages will be marked. The search can be aborted at any time by pressing CTRL-C or the spacebar.

CLEARMARK
: The user is taken to the clear makred files routine that can also be activated from the file menu.

CLEARMARKEDMSGS
: Runs through all conferences and clears the marked messages string. When this command is executed no messages are marked any more. Nothing is returned.

CONTINUE
: Brings up the "'Press RETURN to continue.'" prompt and awaits a response from the user. It takes no arguments and returns nothing.

DOWNLOAD
: executes the download subroutine. PROTOCOLSEND can be used to send files but not the marked files so it is necessary to have a command that allows the download of marked files.

DROPDTR
: closes the serial device which leads to a droped DTR. This terminates the connect if the modem is set to destructuve DTR handling (recommanded)

ECHOMAIL
: Returns 'TRUE' if echomail has been entered this call by the user, or 'FALSE' if not.

```
ECHOMAIL
if RESULT='TRUE' then call export_echo
```

`FEEDBACK` Functionally identical to the function from the Main Menu. Returns `CANCELED` if the mail has not been saved and `STORED` when the message is saved. Returncode is 5 when the message was canceled.

`FEEDMAIL` When a user used feedback this returns `TRUE`.

`FILEAREA` <*area number*>
Changes to the specified file area. If the change was successful, 1 is returned, otherwise 0, indicating that either in invalid area number was given or that the user did not have sufficient access.

`GETCARRIER`
Checks to see whether carrier is present or not. Returns either `TRUE` of `FALSE`. It takes no arguments.

`GETCCHAR` Same as GETCHAR, but returns are case sensitive.

`GETSCHARSTACK`
This command returns the actual command stack of TransAmiga and deletes it. This means you can examine the command stack of TransAmiga and use it for your own programs.

`GETCMD [`<*prompt*>`]`
This command works in much the same way as `GETCHAR`, except that if the user does not have hotkeys selected, it will wait for the user to press `RETURN`. Any surplus characters entered by the user will be kept in his/her command stacking buffer. You can optionally supply a prompt string for it to display before waiting for the command. The command entered will be translated to uppercase automatically. It returns the character received or `###PANIC` in the case of unexpected logoff (due to carrier loss, timeout, etc).

`GETMSGAREAADDR` <*areanumber*>
Returns the net address of the given area. If the value is out of range or the area is no network area (PRIVILEGED/LOCAL) `ERROR` is returned. Otherwise the address is returned in fidonet format as it has been configured in Message.cfg. If no domain is in the config 4D format is used otherwise you get 5D format.

`GETMSGAREATYPE` <*areanumber*>
Returns the type of the specified area. If the value is out of range `ERROR` is returned. Otherwise you will get `LOCAL, MATRIX, ECHO or PRIVILEGED`. See Section 2.4.2 [Messages], page 11 and Chapter 5 [Fidonetting], page 47 for details.

`GOTOBLLTN`
`GOTOEXTRAS`
`GOTOFILE`
`GOTOMAIN`
`GOTOMSG`

`GOTOSET` These commands jump to to the corresponding menues and continue there.

NOTE! This is very tricky and special! Use these command only at the end of your ARexx script! BTW: they only work within a script. The only command after these should be EXIT. Example:

```
GOTOEXTRAS
```

`ICONIFY <ON|OFF>`

As the name says `ICONIFY ON` activates TransAmiga's iconify option and `ICONIFY OFF` disactivates it. Programs that use transamiga.library have not to be aware that they can't print to the window as the library knows itself if it is allowed to use a window. The icon can be double clicked to reactivate the window(s). The same happens if something is droped on the icon. Nothing is returned.

Remark: Make sure your chat program does not use CHR(7) to call the sysop or you will never see a flash/hear the system beep as with active iconify no system displaybeep is executed. Example:

```
ICONIFY ON
```

`INFO <`*keyword*`>`

This is also for compatibility with some standards. <*keyword*> can be one of the following:

`AUTHOR` returns the names of the authors

`BASE` returns the name of the ARexx port although I don't know what use it should have as everyone knows it

`COPYRIGHT`
returns the copyright notice

`DESCRIPTION`
returns a really short description of the program

`SCREEN` should return the name of the screen TransAmiga has it's windows on. As this info is not available with HiSoft BASIC it is not supported at the moment

`TITLE` returns the name of the program

`VERSION` returns the same as VERSION ARexx command.

`JOIN` Takes a user through the process of joining the board, in the exact same way as the Join command from the Main Menu. Takes no parameters and returns 1 on success and 0 on failure.

`LINEMSGS <ON|OFF>`

Switches messages between lines on or off.

`LISTFILEAREAS`

This command lists all file areas the user has access to, two areas per line, with each area numbered. It returns nothing and takes no arguments.

`LISTFILES`

Lists the files in the current file area, giving a total at the bottom. More? prompts will be brought up when necessary, and the user can abort the listing by pressing `CTRL-C` or `SPACE`. It takes no arguments and returns nothing.

`LSTMARKEDFILES <`*path+filename*`>`

Creates a new file with specified parameter. The file lists all the files that are actually marked. One at each line. If no file is marked nothing is done and `FAILED` is returned. If everything worked `SAVED` is returned. Be carefull, it has to be possible to write the file or HiSoft BASIC may display a runtime error so pay attention that there is no lock on a file of the same name and that the path exists. Example:

```
LSTMARKEDFILES Ram:MarkedFiles
```

`LISTMODE <0|1>`

Allows you to select either browse or regular mode for listing files. A mode value of 0 indicates normal mode while a value of 1 switches to browse mode. Returns nothing.

`LISTMSGAREAS`

Lists all message areas the user has access to, two per line with each area numbered. It takes no arguments and returns nothing.

`LISTONLINE`

Displays the current BBS online activity. Takes no parameters and returns nothing.

`LISTPROTOCOLS`

Displays the available file transfer protocols to the user along with their corresponding number. Takes no arguments and returns the number of available protocols.

`LISTUSERS`

Functionally identical to the function from the Main Menu. Returns nothing and takes no parameters.

`LOADLANGUAGE <`*language*`>`

Loads in a new language file, over-riding the current one. The language file is searched for in the configurations directory, and the extension .Lang will automatically be added. Returns nothing.

```
LOADLANGUAGE "francais"
```

`LOCALDISPLAY <ON|OFF>`

`LOCALDISPLAY OFF` turns the output of the terminalwindow OFF. `LOCALDISPLAY ON` naturally reactivates the local display. You can use this as a kind of Iconify. This makes the output to the serial port independant from the local display speed.

ATTENTION: on local lines you may fool yourself when you turn it off as you then can not see what happens :-) Example:

```
LOCALDISPLAY ON
```

`LOCALMAIL`

Returns TRUE when a local mail has been written and FALSE if none.

`MARKFILE` <*filename*>

Marks the given filename if less than 10 have are currently marked. It will scan all file libraries available to the user for the file, it need not be in the current library. It will strip leading and trailing spaces from a filename. Returns the full path to the marked file, `FAIL` if the file could not be found or `ALLREADY` when the file is already marked.

```
QUERY 'Enter file to mark: ' ; file=RESULT
if CARRIER=1 then do
     MARKFILE file
     if RESULT='FAIL' then PRINT 'Not found!'
end
```

`MARKMSG` <*number*>

Marks the given message number in the current conference. Returns nothing.

`MASKOUT` <*conference number*>

Masks out a conference of the given number. Nothing is returned.

`MODEMCMD` <*string*>

Sends the given string to the modem, and will expand the imbedded control code `~` to equal a half second pause, and `|` to equal a carriage return.

```
MODEMCMD '~~~+++~~~ATH|~~'
```

`MORE` Brings up the '`More -Y/N/C-`' prompt and waits for user input. TransAmiga does no checking on what the user enters, it is up to the ARexx macro to do this. It takes no arguments.

`MSGAREA` <*area number*>

Changes to the specified message area. If the change was successful, 1 is returned, otherwise 0, indicating that either in invalid area number was given or that the user did not have sufficient access.

`MOVEFILE` <*file*>\\<*library number*>

Moves a file from the actual conference to the specified conference - if that conference exists. There are several things returned. If the file does not exist `NOFILE` is returned. If the file would be moved to the same are it allreaddy is `SAMEAREA` is returned. If the move worked `SUCCESS` is returned. Any other failtrue returns `FAILED` (e.g. the file library number does not exist). The parameter of this command has a special format. Example:

```
MOVEFILE testfile\\5
 moves the file 'testfile' from the actual area to file library 5.
```

NETMAIL Returns TRUE if netmail has been entered this call by the user, or FALSE if not.

```
NETMAIL
if RESULT='TRUE' then call export_matrix
```

NEWMSGS Displays the number of new messages that have been posted in each area the user has access to since his/her last call. It takes no arguments and returns the number of found new messages. As the User can break this scan you cannot be sure that the number is correct.

NEWFILES Prompts the user at what date to start searching for new files from. It searches all areas and will display new files to the user. It takes no arguments and returns nothing.

NOP NO OPeration, a do nothing command.

PAGESYSOP

Writes to the Status Window and the log that the user has paged, it will not beep. It takes no arguments and returns nothing.

PROTOCOLRECEIVE <*directory*|*filename*>

Receives a binary file using the current file transfer protocol. For batch protocols, you provide a directory, for non-batch protocols, a full pathname. It returns a list of the files received, enclosed in quotes, each file separated by a space.

PROTOCOLSEND <*files...*>

Sends the given files using the current file transfer protocol. The full-path name of the file must be given, TransAmiga will not search for the file in the file areas. Note that although several files can be passed, some protocols are non-batch and will only send the first file. Also note that a user's current protocol may be unpredictable unless the SETPROTOCOL command has been used. PROTOCOLSEND now *should* be able to send more files than maxfiles. However 19 is the maximum that should be not exceeded or TrXPR will lock the system when it's called.

QUERY [<*prompt*>][\\<*default*>]

This command waits for input from the user. The user may enter up to one line of text, and must terminate the line by pressing RETURN. If a prompt is supplied, this will be displayed immediately before waiting for input. A default for user input can be specified by having two backslashes (\\) immediately after the prompt text, followed by the desired default input. The default input will be displayed for the user to edit. The string entered is returned.

```
QUERY 'What planet do you live on: \\Earth'
planet=RESULT
```

`QUERYY [<prompt>]`

This will prompt the user for a Yes or No response, with the default being Yes. If a prompt is supplied, it will be displayed immediately before waiting for user input. TransAmiga translates user input to upper case, but does nothing else. The character entered by the user is returned.

```
QUERYY 'Are Amigas good computers?'
if RESULT='Y' then smart=1 else smart=0
```

`QUERYN [<prompt>]`

This is exactly the same as the above, except that the default response it No.

```
QUERYN 'Is the sky green?'
if RESULT='N' then smart=1 else smart=0
```

`RAISEDTR` couterpart to `DROPDTR`. Just reopens the serial device which leads to a raised DTR. You can use the two commands to let TA release the serial device if a program does not know about shared mode. ATTENTION: In this case you have to set the modem to non-destructive DTR operations.

`READMODE <mode>`

Selects the read mode for reading messages. Current defined modes are 0 for read all available messages, 1 for reading personal mail only, and 2 for reading marked messages only. The read mode is always set as 0 when the user logs on. Returns nothing.

`READMARKEDMSGS`

This cycles through all message conferences that the user has access to and the marked messages. This can be useful as a means of immediately displaying messages to the user that have been found in some kind of search.

`READNEWMSGS`

Starts reading new messages in the current area, equivalent to the command from the message menu. Takes no parameters and returns nothing.

`RELOGIN` As the name states the user is logged off and presented the login sequence without dropping the carrier. The logoff of the user is just virtual. This means TransAmiga resets everything that is needed and saves the userfile. Then TransAmiga jumps to the connect-part. It restarts at ANSI DETECTION. Again ATTENTION! THIS HAS TO BE THE LAST COMMAND before an EXIT command and only works within an ARexx script.

`RENAMEUSER <new name>`

Renames the user to the given new name. Saves a new userfile, corrects HANDLE if there was one and deletes the old userfile. *Attention:* You have to check the username for unallowed characters yourself! Don't blame me for gurus caused by your missing check.

`RESCAN` Rescan the message areas for the high and low message pointers. This should be used if your ARexx macro has run TrUMP to adjust the message pointers, this makes sure TransAmiga knows about these changes.

`RUNDOOR <`*cmd*`>`

Starts a door using the given command. Before the door is saved, a TransAmiga v1.0x format door information file is saved in T:, the name of which can be passed using %f (see below). Several imbedded control codes are supported:

`%b`	Baud rate of the caller
`%n`	BBS line number
`%f`	Name of door info temporary file
`%c`	BBS master configuration file for this line

The command's result code is returned, or '-1' if the command could not be executed.

```
RUNDOOR 'BBS:Extras/MyDoor/MyDoor -n%n -c%c'
```

NOTE: This command is only useful some doors, particularly older ones designed for TransAmiga v1.0x, and doors that make minimal use of TransAmiga.library. Most doors (such as '`TrMaint`' and '`TrShell`') will need to be started up with the built-in ARexx command `"address command <cmd>"`. For an example of starting doors this way, see the examples of starting '`TrMaint`' and '`TrShell`' from the BBS, and you should also look up "address command" in your ARexx documentation.

Hopefully the door's documentation will clearly state how it should be started, but when in doubt, use the "address command" method.

`SAVEMSG <`*filename*`>\\<`*msg number*`>`

This saves the indicated message number to the given file. Note that no checking is done to see if the user should be allowed to read this message. The filename must be immediately followed by two backslashes (\\) followed immediately by the message number. Both arguments are required, nothing is returned.

`SAVEUSERFILE`

If the user has not joined till now a new userfile is created similar to the join sequence. He gets a new usernumber and is written to NewUsers.BBS. If he already joined the BBS, the userfile is just saved. In both cases the user's access has to be higher than 0 or nothing is done.

`SCREENTOBACK`

Puts TransAmiga's screen behind all others. Nothing is returned.

`SCREENTOFRONT`

Brings the screen TransAmiga's line has it's terminal window on to the front in any case. You can use this e.g. if you want TA's screen to come to front when your chat

script is run. To use this, start TransAmiga with the -p option (see Section 3.1 [Running TransAmiga], page 17). Only available for OS2.x (>v36).

`SEARCHFILES` <*search string*>

Searches all file areas that the user has access to for a file that has the given string in its filename or description, and displays those files to the terminal. Returns `FOUND` if new files where found and `NONE` if none are found and in this case return code is set to 5.

`SENDMENU` <*filename*>

This is very similar to the `SENDFILE` command. The only differences are: the file is assumed to be in the BBS text directory (therefore, you should not pass a full pathname); the file may be interrupted when it is being displayed by a the user pressing a key, which is taken to be a menu command. This command can then be retrieved using any input commands such as `GETCHAR`, `GETCMD`, etc.; and TransAmiga will also attempt to see if there is a menu for the user's access level, in exactly the same way as it does when displaying menu files.

`SENDNODEMSG` <*line number*> <*text*>

Sends *text* to the given line as online message. *text* should not exceed two lines and you have to insert the `CR` yourself. Only one `CR` is allowed and don't use it in front or at the end of the text. Basically the same as the command in the main menu.

`SETACCESS` <*access*>

Changes the online user's access level to the given value. Returns nothing.

`SETALIAS` <*name*>

Changes the online user's alias to the given value. Returns nothing.

`SETBIRTH` <*date*>

Changes the online user's birthdate to the given value, which must be in YYMMDD format. Returns nothing.

`SETBYTERATIO` <*value*>

Sets the kbyteratio to the specified value. Maximum for this value is 2^31-1. 0 disables Ratio. Example:

```
SETBYTERATIO 10240000
  User's byteratio is 10000 Kb from now on.
```

`SETCHARSTACK` <*string*>

You can use this command to clear the command stack when you use an empty string ("") or set it to anything you want. In Conjunction with the above command you can examine the command stack and react as your program should and then set the command stack to the string that continous after your program. Example:

```
SETCHARSTACK +B1
```

`SETCLEAR <ON|OFF>`

Activates clearcodes if `ON` is specified and sets the user's value to it. `OFF` naturally has the opposite effect. Returns `FAIL` if something else is specified and `SET` if the value is ok.

`SETCOMMENT <`*comment*`>`

Changes the one-line comment to the given comment, and attributes it the current user (or to his/her handle if the user has one). Returns nothing.

`SETCITY <`*city*`>`

Changes the online user's city to the given value. Returns nothings.

`SETCOMP <`*computer*`>`

Changes the online user's computer type to the given value. Returns nothing.

`SETCREDIT <`*credits*`>`

Sets the current user's netmail credit.

`SETECHOMAIL <0|1>`

This command sets the value `ECHOMAIL` will return. 1 is `TRUE` (echomail was written) and 0 is `FALSE` (No echomail was written). Nothing is returned.

`SETEDITOR <`*editor type*`>`

Sets the type of editor for the online user. Values allowed currently are 0 for the internal line-editor, and 1 for the external full-screen editor. Note that TransAmiga does no checking here as to whether the user has ANSI active or not, so beware of this when setting people to the full-screen editor. Returns nothing.

`SETFEEDMAIL <0|1>`

This command sets the value `FEEDMAIL` will return. 1 is `TRUE` (feedback was written) and 0 is `FALSE` (No feedback was written). Nothing is returned. Remark: The terminal window's title is changed to tell the sysop that feedback was written. This info gets lost when something else changes the wondow title again. (e.g. logoff).

`SETFLAGS <`*flags*`>`

Changes the access flags of the online user. Should be eight characters long with an X indicating the flag is set and a - indicating it is unset. Returns nothing.

`SETGRAPHICS <`*graphics type*`>`

Changes the graphics type of the online user. Values allowed currently are 0 for ASCII, and 1 for ANSI. Returns nothing.

`SETHIMSG <`*number*`>`

Sets high message value to `<`*number*`>`.

`SETHOTKEYS <`*boolean*`>`

Changes the status of hotkeys for the online user. 0 turns them off, 1 turns them on. Returns nothing.

`SETLASTREAD <`*number*`>`
: Changes the user's last read pointer in the current message area to the given number. Returns nothing.

`SETLENGTH <`*length*`>`
: Changes the screen length of the online user to the given value. Returns nothing.

`SETLOCALMAIL <0|1>`
: This command sets the value `LOCALMAIL` will return. 1 is `TRUE` (local mail was written) and 0 is `FALSE` (No local mail was written). Nothing is returned.

`SETNETMAIL <0|1>`
: This command sets the value `NETMAIL` will return. 1 is `TRUE` (netmail was written) and 0 is `FALSE` (No netmail was written). Nothing is returned.

`SETPHONE <`*tel. no.*`>`
: Changes the phone number of the online user to the given value. Returns nothing.

`SETPOSTAL <`*postal code*`>`
: Changes the postal (or Zip) code of the online user to the given value. Returns nothing.

`SETPROTOCOL <`*protocol number*`>`
: Changes the protocol of the online user to the given protocol number and stores it.

`SETPROV <`*province*`>`
: Changes the province (or territory or state) of the online user to the given value. Returns nothing.

`SETPSWD <`*password*`>`
: Changes the password of the online user to the given value. Returns nothing.

`SETRATIO <`*max. downloads*`>`
: Changes the maximum number of files the user may download for each upload to the given value, which may be from 1 to 255. Returns nothing.

`SETRIP <0|1>`
: Sets the status of user's RIP Script setting. 0=RIP off, 1=RIP on. Returns nothing.

`SETTIMELIMIT <`*minutes*`>`
: Sets the current user's daily time limit, but does *not* change his/her time remaining for this call.

`SETUSERMODE <EXPERT|NOVICE>`
: Switches menu display on (`NOVICE`) or off (`EXPERT`).

`SETSTREET <`*street*`>`
: Changes the street address of the online user to the given value. Returns nothing.

`SETTIME <`*minutes remaining*`>`
: Changes the amount of online time the online users has for today. Returns nothing

`SETUSERCOMMENT <string>`

Stores a sysop comment in the userfile when `SAVEUSERFILE` is used or the user loggs off. As soon as TA receives this command the window title is updated if *<string>* is not empty. When the user logs on next time the terminal window's title will be changed to this string. Nothing is returned.

Remark: The name of the user is not automatically added. If you want to know who is online you should use the name in the string.

Example:

```
SETUSERCOMMENT Mr. Elite is so ugly.
```

`SETUSERNAME <string>`

Changes the username to *string*. Remember the restrictions to the characters used in the username or you may have some trouble :-)

`SINGLERESCAN`

Rescans the actual message area. No parameters, no special return value.

`STOREMSG <filename>\\<subject>\\<touser>[\\<address>]`

The part in [] is only needed in echo/netmail areas. In local areas it is ignored. This command stores the textfile 'filename' in the actual conference as a new mail. 'filename' is the compleate path and name of the textfile. As 'subject' indicates it's the subject of the message. 'touser' is the person the message should be send to. 'address' is a fidonet compatible addess. There are no real checks if the address is correct so be sure it is! It was a lot of work and I hope I thought about every aspect. Nothing is returned. Remark: not fully tested Example:

```
STOREMSG ram:testmsg\\Testmail\\Sami Radwan
```

`SYSTEMINFO <character>`

This returns the piece of system information corresponding to the given character. The characters available are the same for the system information imbedded control codes that area available in text files, and the same information will be returned. Use both this and the `USERINFO` command to find out information about the system setup and the current online user.

An example of its usage would be to send 'SYSTEMINFO 7 ' to check the access of the current file area, to see if the user has access.

```
SYSTEMINFO b ; last_caller=RESULT
```

See Section 4.2.5 [Imbedded Control Sequences], page 39.

`TEXTEDIT` Calls the appropriate editor, depending on the user's editor choice, and whether or not it is a local logon. It returns the name of the file that contains the text the user has entered, or `###ABORTED` if the user aborted the editor. Takes no parameters.

```
TEXTEDIT ; file=RESULT
if file~='###ABORTED' then
     address command 'copy' file dir||'signature'
```

`TIMESTATS`

Returns the same as the command from the main menu.

`UNMARKFILE` <*filename*>

Unmarks a previously marked file. If the file could be unmarked it returns `UNMARKED` if it couldn't be unmarked `FAIL` is returned. Example:

```
UNMARKFILE test.lha
```

`USERINFO` <*character*>

This returns the piece of user information corresponding to the given character. The characters available are the same for the user information imbedded control codes that area available in text files, and the same information will be returned. Use both this and the `SYSTEMINFO` command to find out information about the system setup and the current online user.

An example of its usage would be to send `USERINFO x` to check the ANSI status of the user, and set some ANSI variables accordingly.

```
USERINFO a ; name=RESULT
USERINFO 0 ; acc_level=RESULT
```

See Section 4.2.5 [Imbedded Control Sequences], page 39.

`VERSION` Returns the version number of TransAmiga running on that line. It's just for ARexx compatibility with some standard.

`VERSIONINFO`

Displays the version information to the user. It takes no arguments and returns nothing.

`VIEWARC` <*archive*>

Display the contents of an archive in one of the file areas. Same as the command in the file menu.

`VIEWMSG` <*number*>

Displays the given message number, from the current message area. Last read pointers are not updated. Returns a non- zero value upon failure.

`WINDOWTOBACK`

Puts TransAmiga's terminal window behind all other windows on that screen. Nothing is returned.

`WINDOWTOFRONT`

Puts TransAmiga's terminal window in front of all others on the screen. Nothing is returned.

`YELL` Flashes the screen 5 times and posts a message in the status window stating that the online user has paged. Returns nothing and takes no parameters.

6.5.4 Commands Recognized with no Caller Online

Here are the commands that TransAmiga understands when it was started with the `-r` option, these commands are not recognized when there is a caller online:

`BAUDRATE` <*baud*>
: Specifies the baud rate the TransAmiga should log the caller on at. Typically this is followed closely by a `LOGON` command. Equivalent to the `-b` command line parameter. It will only actually change the baud rate of the serial port if the `LOCKBAUD` or `-l` options were not used. It returns nothing.

`LOCAL [WAIT]`
: Does a local logon, equivalent to the Local Logon item from the pull-down menu. It takes no arguments and returns nothing. If the optional `WAIT` keyword is passed, this command will not returned until the local user has logged off, otherwise, it will return immediately.

`LOCKBAUD` <*baud*>
: Specifies at what baudrate TransAmiga should lock the serial port at. Equivalent to the -l command line parameter. It returns nothing.

`LOGON [WAIT]`
: Tells TransAmiga to log on an online user. Typically a combination of `BAUDRATE` and possibly `LOCKPORT`, `MAXTIME` and `USERNAME` commands are issued right before this. It takes no arguments and returns nothing. If the optional `WAIT` keyword is passed, this command will not return until the call has completed, otherwise it will return immediately.

`LOGONPSWD` <*password*>
: <*password*> is checked while the normal login sequence (initiated with `LOGON` ARexx command). If the check fails the normal password routine is activated. Nothing is returned. Remark: Part of EMSI Login support Example:

```
LOGONPSWD SECRET
```

`MAXTIME` <*minutes*>
: Specifies the maximum amount of time the next caller will have. Equivalent to the `-t` command line parameter. It returns nothing.

`MDMCONNECT` <*modem connect message*>
: This ARexx command is for TrapDoor v1.84. TrapDoor has %C as option that returns the connect message of the modem. <*modem connect message*> should be replaced by that string. Nothing is returned. Example:

```
MDMCONNECT CONNECT 14400/ARQ
```

`RECONFIG` Does the same as the pull-down menu command 'Reload Config'. It closes the window(s) [/screen] frees some memory and starts the setup with reading configfiles.

`RESCAN` Tells TransAmiga to rescan the message areas. You will need to do this if TrUMP (see Section B.17 [TrUMP], page 113) has been run because of changes to the message pointers (eg. after a renumber or import) without the BBS being restarted. It takes no arguments and returns nothing.

`QUIT` Simply tells TransAmiga to shut down completely, equivalent to the Quit item from the pull-down menu. It takes no arguments and returns nothing.

`UNLOCKPORT`

Tells TransAmiga not to lock the serial port, and to adjust the baud rate of the serial port to that of the caller. This is the default, and is only used to over-ride a `LOCKPORT` or `-l` command.

`USERNAME <`*user*`>`

This can be used to bypass the initial user name prompt once the `LOGON` or `LOCAL` command is executed. The logon process will start with the BBS evaluating the name entered. If the user is currently a member, the password prompt will be entered. If not, the guest user process will take place. If its an invalid name, then the BBS will display the user prompt.

This command is very useful with frontends that take input from the user (eg. Welmat). This way you can get the user's name from the frontend and pass it directly to the BBS.

The commands are typically used in conjunction with some kind of frontend, such as a FidoNet mailer. Using these commands, you can have the BBS in memory all the time, and have the frontend run an ARexx macro to "wake-up" the BBS and save the usual waiting period while the BBS loads, making the transition from frontend to BBS almost instantaneous for the caller.

6.6 SkyLine/C-Net ARexx Emulation

TransAmiga also understands many of the same commands as SkyLine BBS does. Those of you familiar with its command set will have noticed that several commands are the same (eg. `GETCHAR` and `QUERY`), while other commands work are very similar to SkyLine BBS ARexx commands, but have different names (such as `SYSOPLOG`, which is directly equivalent to SkyLine's `LOGENTRY` commands). TransAmiga therefore provides synonyms for these commands, so that most SkyLine ARexx macros can be run under TransAmiga almost unmodified. All that needs to be done is to remove the line:

```
address 'SkyLine'
```

from any ARexx macros, and they should work under TransAmiga. If they do not, it is because they are using SkyLine ARexx commands that TransAmiga does not support. If this happens, usually only very simple modifications will be necessary to support the equivalent TransAmiga commands.

In addition to TransAmiga's commands which are already compatible with SkyLine, TransAmiga supports: `TRANSMIT`, `SENDSTRING`, `LOGENTRY`, `BUFFERFLUSH`, `SHUTDOWN`, `CHECKIO`, and some instances of the `GETUSER` command.

6.7 Example ARexx Macro

To help you get started writing your own ARexx macros for TransAmiga, a very simple example is given here. Its not particularly useful for any purpose other than to illustrate how to use TransAmiga ARexx interface. Lines have been numbered in this manual for easy reference, but actual ARexx macros, naturally, would not have these numbers.

```
 1: /* Example.trans - the World Famous Hi-Lo Game! */
 2: options results
 3: userinfo a ; username=result
 4: print "Welcome" username "to The World Famous Hi-Lo Game!"
 5: newline
 6: keepgoing=1
 7: do while keepgoing=1
 8:     num=random(1,100,time(e))
 9:     print "OK, I have chosen a number between 1 and 100, and"
10:     print "it is your job to guess what it is."
11:     notright=1
12:     guesses=0
13:     do while notright=1
14:          guesses=guesses+1
15:          prompt 3 'NORMAL' '"Enter guess #'guesses': "'
16:          guess=result
17:          if carrier=0 then exit 10
18:          if guess<num then print "Too low."
19:          if guess>num then print "Too high."
20:          if guess=num then notright=0
21:     end
22:     print "You win!  And it only took you" guesses " tries!"
23:     newline
24:     prompt 1 'YESNO' '"Would you like to play again?"'
```

```
25:     answ=result
26:     if carrier=0 then exit 10
27:     if answ='N' then keepgoing=0
28: end
29: print "Bye!"
30: exit
```

line 1: This is a comment line, every single ARexx macro must start off with a comment line like this. Comments in ARexx are enclosed in /* and */

line 2: All your TransAmiga ARexx macros must have this line. It tells ARexx that it should expect the host (that is, the BBS) to set the result after a command.

line 3: This is the first usage of a TransAmiga ARexx command, `USERINFO`. `USERINFO` takes as a parameter a code which tells it what piece of information to return. In this case, the code is A, which means it should return the current user's name. Our ARexx macro takes this value (all values returned by TransAmiga are stored in the variable result) and copies into a variable called username (since the result variable will be changed by other commands).

line 4,5: These two lines make use of the `PRINT` and `NEWLINE` TransAmiga ARexx commands. Note in line 4 how you can easily include variables in the arguments you pass to TransAmiga ARexx commands.

line 6,7: This initiates the play loop for our game, using ARexx's `DO WHILE`... construct.

line 8: Makes use of ARexx's built-in `RANDOM()` function to generate the number that the user is going to have to guess.

line 9,10: Uses TransAmiga's `PRINT` command again to show the user some information.

line 11,12,13: Initiates the guessing loop, again using ARexx's `DO WHILE`... construct.

line 14: Increments the guesses variable that we initialized earlier, to keep track of the number of guesses the user has made.

line 15,16: Prompts the user for a guess using TransAmiga's `PROMPT` command. `NORMAL` indicates the input type, and 3 is the maximum number of characters the user can enter (since the number is between 1 and 100, there's no point in entering anything longer than 3 characters). The text the user entered is then copied into the variable guess.

line 17: This is a very important line. While our macro was waiting for input, the user could have dropped carrier, had the idle timer expire, or had his/her time limit run out. In which case, it would be pointless and stupid for our game to continue without a user there to play. What we do then is check the TransAmiga variable CARRIER to see whether or not the user is still there. If it's 0, then he/she is gone, so our macro should exit. Otherwise, we can continue.

line 18,19,20: Here we check the user's guess to see if it is too high or too and tell the user so. If the user's guess is correct, we exit the guessing loop.

line 21: The end of the guessing loop, go back up to the beginning of this loop.

line 22,23: Congratulate the user using TransAmiga's `PRINT` and `NEWLINE` commands.

line 24,25: Ask the user if he/she would like to play again. This uses TransAmiga's `PROMPT` command again, but this time we use the `YESNO` input type, since we want the user to answer Yes or No. The user's response (which is guaranteed to be either Y or N) is copied into the variable answ.

line 26: Once again, the all important check to make sure the user is still with us.

line 27: If the user responded with No, then exit the playing loop.

line 28: End of the playing loop, go back to the start.

line 29: Say good-bye!

line 30: Exit, end of our macro.

Any actual macro you were to write would naturally be a little more involved that this. For example, you may want to display ANSI graphics to users that have ANSI turned on. Also, a little more "idiot-proofing" should be done (for example, if the user enters text instead of a number, the macro will halt with an error). However, this should give you a good idea of what a TransAmiga ARexx macro looks like and you should be ready now to write some of your own.

7 Politics

7.1 Disclaimer

No warranties are implied or expressed with regard to the fitness or merchantability of TransAmiga BBS for any particular purposes. All risks and damages, incidental or otherwise, arising through the use or misuse of TransAmiga BBS are entirely at the responsibility of the user.

While considerable effort has been made to provide you with a reliable product, there is no guarantee that the programs are 100% "bug-free". Any bugs will be fixed at the author's discretion in possible future releases of TransAmiga BBS.

7.2 License

"TransAmiga" refers to a set of programs for Amiga computers by Sami Radwan (see Section 8.4 [Author], page 91) making up the TransAmiga BBS package.

1. TransAmiga is the copyrighted material of Sami Radwan. It may only be used in accordance with the conditions set out by this license agreement.
2. So that you may evaluate TransAmiga, you are granted a limited use license for a period of twenty (20) days after initial usage. After this period you must register with the author or discontinue using TransAmiga.
3. Registration entitles you to use TransAmiga and any future 1.x versions of TransAmiga as long as you wish, subject to any special licensing conditions attached to future versions.
4. Sami Radwan is in no way obliged to provide future versions or support for TransAmiga.
5. You may not modify, disassemble, decompile, re-source or otherwise reverse engineer TransAmiga.
6. You may resell an original file '`TransAmiga.Key`' when following some simple steps. see Section 7.4 [How to Register], page 86 for details.
7. Distribution is limited to the original shareware archive file only, none of the member files may be omitted and no new additional files may be included. Furthermore, no profit or other material gain may be realized for distributing TransAmiga with the sole exception of that which is required to cover the cost of the medium. Bundling TransAmiga, the distribution archive, or any member files, with any commercial product is not permitted without the express

written consent of the author. At no time are you permitted to distribute or modify the file '`TransAmiga.Key`'.

8. TransAmiga may not be used in any unlawful or illegal manner.

7.3 Shareware

TransAmiga is distributed under the concept of shareware. Some people like to call this particular brand of shareware "crippleware" because I am distributing a version with several features removed. It's not "crippled" though, because enough features are enabled for you to set up the BBS, use it, and decide what you think about it.

This allows you to "try before you buy." If, after a period of 20 days you decide that TransAmiga is not the thing for you, simply delete it and all it's files, and no harm is done. However, if you decide that you want to keep using it, you simply register yourself, and you get your own keyfile making the BBS fully functional.

7.4 How to Register

To register, choose the appropriate registration form you got with this package and follow the included instructions.

Generally, the registration fees are as follows:

Registration fee for Germany: DM 55.00
Registration fee outside Germany: DM 60.00

Further details can be found in the registration forms.

There are several registration sites outside Germany.

See Section 8.2 [Support], page 89 for details.

An original '`Transamiga.Key`' may be resold when following these steps:

1. The previous owner has to notify Sami Radwan (see Section 8.4 [Author], page 91) about the new owner.

2. The new owner can get a new, personalized keyfile from the Author for DM15,- via usual mail on a floppy disk containing the new keyfile and the current release version or for DM10,- via electronic mail (email, Fido, AmigaNet, trekuk, TNGnet) the new keyfile only.
3. A complete registration_form has to be send to the Author by any means.

8 Conclusion

8.1 Future

Here are some things planned for Version 2.0:

- Completely rewritten in Oberon (no more HiSoft Basic crap :-))
- new Client/Server system
- extremely expanded ARexx port
- new powerful config editor with a nice GUI
- complete RIPscrip support
- freely definable menu system
- direct support for 'UMS', the Universal Message System for the Amiga
- completly usable installation via Installer
- many things more ('configurability 'til you drop dead but still easy to adjust to your needs)

Ed's note: I think this version will be a little revolution in the Amiga BBS software section and I am eagerly waiting for a first glimpse at it :-)

8.2 Support

As a registered user, you will be entitled to support from the author from the following sources:

Amiga on Phone
TransAmiga developement & registration
Official German TransAmiga Support Board
Place: Karlsruhe
SysOp: Sami Radwan
Phone: +49-(0)721-859432 and +49-(0)721-859433. Both are served by US Robotics dual standard modems (v.fc/v.34/v.34+).
Fido: 2:2476/12
e-mail: rasa0011@FH-Karlsruhe.DE

Amiga Something
Official NL TransAmiga Support Board
SysOp: Jascha Hoogenraad

Place: Den Haag
Phone: +31-70-3462376 (v32bis/v42bis)
AmigaNet: 39:153/107
Fido: 2:281/541.3

Amiga Time
Official American TransAmiga Support BBS (Head Quarter)

Attention: Amiga Time is gone offline and no longer Support BBS and North American Registration Site!
Please contact the Author about a new Support BBS.

Crazy Diamond BBS
registration site
Official Australian Support Board
SysOp: Chris Quonoey
Place: Oakleigh Vic
Phone: +61-3-9570-2303 (v34)
Fido: 3:633/359

GateWAY BBS
registration site
Official UK Support Board
SysOp: Neil Barrett
Place: Grays Essex
Phone: +44-1375-393816 (v32bis/v42bis)
Fido: 2:257/99

The Last Tango
Official UK Support Board
SysOp: Tom Pereira
Place: Twickenham
Phone: +44-181-241-6442 (v34)
Fido: 2:254/270

TransAmiga may be periodically upgraded. All upgrades will be made available on the support boards, for ftp via Aminet and on the Amiga Distribution System's (ADS) dedicated TransAmiga area, ADSTRANS. Since registered and unregistered users actually use the same program, there is no need to mail registered users upgrades. New versions will recognize your keyfile and run in registered mode. When new versions come to gamma testing stage, they can be obtained from the TransAmiga support boxes by registered sysops only.

NOTE: gamma versions may only be distributed by the TransAmiga support boards and only registered sysops are allowed to use them!

To get the latest version of TransAmiga via Internet, use

www: `http://www.winnipeg.freenet.mb.ca/~lss133/ta/ta.html`

ftp: `hp48.rhein-neckar.de` `directory` `pub/AMIGA/TA`

There are also several echomail conferences available via FidoNet dealing with TransAmiga BBSing:

TRANSAMIGA an international conference which should be available almost everywhere and which is also on the north american Fido backbone

TRANSAMIGA.GER
the german pendant to TRANSAMIGA, only available in Germany

You can get these echos at least from all the support boards and they should be able to tell you where you can get them in your area.

There also exists a file echo called TAFILES which is distributed to the support boards and linked nodes. Here you can find new TransAmiga releases, tools and other things related to TransAmiga.

8.3 Comments

If you have anything you would like to tell me about TransAmiga, I encourage you to speak. Comments, criticisms, suggestions, praise, bug reports, and the time of day are all very welcome. I really like to receive comments.

See Section 8.4 [Author], page 91 on how to contact the author.

8.4 How to reach the Author

Sami Radwan, the author of the fantastic TransAmiga BBS package, can be reached via:

FidoNet 2:2476/12 or 2:2476/13

e-mail rasa0011@fh-karlsruhe

BBS +49-(0)721-859432 and +49-(0)721-859433. Both are served by US Robotics dual standard modems. Actually in update phase to v.fc.

snail-mail Sami Radwan
TransAmiga Developement
Uhlandstr. 10
D-76135 Karlsruhe
Germany

8.5 Acknowledgements

Special thanks to the following people (in no specific order):

- Special thanks to my parents for all their help.
- Special thanks to Patrick Langer for updating and improving the english docs.
- Special thanks to HiSoft and Maxon for great support and more.
- Many thanks to Mathias Supp for Filelister and a lot more.
- Many thanks to all TransAmiga support sysops and special thanks to the registation sysops.
- Thanks to Timothy J. Aston for starting TransAmiga
- Thanks to Holger Scherer.
- Thanks to Yves Rausch for SMS and other.
- Thanks to Kai Bolay for explanation of a mistake in CASE command and several other tips.
- Thanks to Peer Hasselmeyer and Michael Hettenbach for Foozle.
- Thanks to TrapDevelopment for their fine products.
- Thanks to Commodore for C= Amiga line.
- Thanks to C= UK and CEI blocking each other to buy C= (I will never understand them).
- Thanks to Epson for my Stylus Color, it's really a fun!
- Thanks to VillageTronic for Picasso/Pablo and their help.
- Thanks to Volker Voigt for testing online.
- Thanks to my BBS users for their patience.
- Thanks to Quantum for really fast hard drives. :-)
- Thanks to US Robotics for the Sysop deal and good support.
- Thanks to all my beta testers who told me bugs and other bad things...
- Thanks to Mark Rogers and Iain Black for reporting bad doors.

- Thanks to all registred Sysops.
- Thanks to my friends beeing so patient.

Finally some words of rememberance for Bavin Clarke, who passed away too soon. He was a good support for the TA sysop community and did also the proofreading for the first versions of this manual.

8.6 On Your Own

That's it, folks, you're on your own now. I hope that this document did what is it's purpose: to give you aid and assistance and not to leave you alone and confused by your problems. I tried to make it as logical and easy to understand as possible but still enabling you to use TransAmiga's full power. As Sami was too busy working on the program, I volunteered to write the new docs. They are now completely rewritten, put into new formats and I hope you like them. If you have any suggestions regarding the docs, just tell me.

Patrick Langer, 13 June 1996

email	planger@juglers.gp.in-ulm.de
Fido	2:2487/9533.0
AmigaNet	39:173/30
BBS	++49-7161-582219

9 Appendix

Appendix A Examples

Examples:

This appendix provides numerous examples to help you getting your system running. The examples cover a wide range of applications, and you will probably only need to refer to a few of them depending on how you have things set up. Be careful, as some lines have had to be wrapped so as to fit in the manual.

Sample TrapDoor Configuration: This is an example configuration file for the TrapDoor FidoNet mailer, running as a node.

In addition to the TrapDoor program itself, the TrapDoor package comes with three other programs you will be need in your setup, TrapTell, TrapPoll and TrapList. TrapTell won't be used in our examples, but TrapPoll will, and you will need TrapList, as that is TrapDoor's special nodelist processor. Copy all of the files into Mail:. TrapDoor comes with a shared library called traplist.library, which must be placed in your Libs: directory. There are also a few other tools that come with TrapDoor which you may find useful, but will not be explained here.

Here is a sample configuration file. Comment lines start with a semi-colon (;).

```
; This is your node number.  If you haven't yet received
; yours, use zone:net/9999.o
NODE 1:247/9999.0
; This is the name you've chosen for your system
NAME "TransAmiga Development"
; This should be easy enough, your name
SYSOP "Sami Radwan"
; When you are going to perform password protected session,
; this is where the password goes.  For now, leave at ""
PASSWORD ""
; Your maximum and minimum baud rate.  Unless you're using
; an HST, leave NOLOCK as is.
BAUD 2400 NOLOCK MINBAUD 300
; The address and phone number of your mail feed.
BOSS 1:247/9999.0
CALL 1:247/99
; These parameters define the type of mail sessions TrapDoor
; will do.  Generally you don't have to alter them.
DIETIFNA ZEDZAP
WAZOO
; Your Outbound, Inbound, and Nodelist directories.  If you
```

```
; don't have a nodelist, use NODELIST ""
OUTBOUND "Mail:Outbound"
INBOUND "Mail:Inbound"
NODELIST "NodeList:"
MAXBAUD
RINGS 1
; Some BBS setup commands, leave these as is for now
BANNER "This is my BBS running TransAmiga"
; The BBS startup command
BBSCOMMAND "rx StartBBS %b %B"
BBSINOUT ""
BBSMODE SPAWN
; Log file setup
LOGFILE "ram:TrapDoor.log"
LOGLEVEL 0:2
LOGLEVEL 1:2
LOGLEVEL 2:2
LOGLEVEL 3:2
LOGLEVEL 4:2
LOGLEVEL 5:2
LOGLEVEL 6:7
LOGLEVEL 7:2
LOGLEVEL 8:5
LOGLEVEL 9:2
WRAPLINES
SLOWMODEM 10
; Your modem commands.  On a typical 1200 or 2400 baud modem
; these should be enough
MODEMINIT "ATZ|~~~~~"
MODEMHANGUP "||^|"
MODEMDIALPRE "ATDT "
MODEMDIALPORT "|"
MODEMANSWER "ATA|"
SHARED
; If you've got ARexx, you can use the FRequest.rexx macro
; to allow other FidoNet systems to request files from you.
; If you don't have ARexx, change this to FREQUEST ""
FREQUEST "rx FRequest %i,%o,%l"
; Import.rexx should be an ARexx script that does your
; importing and renumbering
AFTERSESSION "rx Import"
REXXNAME "TrapDoor"
SHOWREXX
DSR
SERIALNAME "serial.device" SERIALUNIT 0 SERIALFLAGS 0
ADJUST -11
LOGWINDOW 30/12/580/125 STATWINDOW 30/139/580/61
SCREENMODE TRAPDOOR
```

```
NOBACKGROUND AUTOOVERSCAN
COLOURS $a00/$fff/$000/$99e
ACCOUNTING ACTMAX "3000 20 2 20 5 2"
ANSWER
RETRIES 1 REDIALDELAY 1
NOQUIET NOSWEPULSE
FKEY "1:EMSI"
FKEY "11:NOEMSI"
FKEY "2:WAZOO"
FKEY "12:NOWAZOO"
FKEY "3:ZEDZAP"
FKEY "13:NOZEDZAP"
FKEY "4:DIETIFNA"
FKEY "14:NODIETINFA"
FKEY "5:BBSMODE SPAWN"
FKEY "15:BBSMODE ZMH"
FKEY "10:SPAWN \"TransAmiga -x\""
```

Save your configuration in Mail: as TrapDoor.Cfg. To start TrapDoor, you would just enter:

```
run TrapDoor
```

StartBBS.rexx:

To start the BBS from TrapDoor, the BBSCOMMAND is executed, `rx startbbs %b %B` in the above example. Here again the needed entries for TrapDoor and GMS:

```
; Entries for TrapDoor to start TA in SPAWN mode
BBSCOMMAND "Rx Rexx:StartBBS %B %b"
BBSINOUT ""
BBSMODE SPAWN

; Entries for GMS to start TA in SPAWN mode
BBSMode=Spawn
BBSCommand="Rx Rexx:StartBBS %B %b"
DropBBSSecs=20
```

Here's an example how this '`StartBBS.rexx`' could look like:

```
/* Start TransAmiga from TrapDoor or GMS */
options results
address TRANSAMIGA.0    /* Adjust this to your needs   */
parse arg rate real
BAUDRATE real; /* Pass Baudrate to Transamiga */
```

```
LOCKBAUD rate; /* Pass Portrate to Transamiga */
LOCKPORT; /* and lock Port at this rate  */
LOGON WAIT /* Will log user on and wait till he logs off */
exit
```

This example should be self-explanatory.

Aftersession:

After the user logged off and TransAmiga has returned control to TrapDoor, there are several things to do. AFTERSESSION accomplishes this task and executes a script or command. In the above example it's `rx import`. Here's an example how this ARexx script could look like if you use TrapDoor together with Foozle as a mail processor:

```
/* Aftersession-Script for Trapdoor */
/* Tell Foozle to import and export */
address 'Foozle' 'IMPORT EXPORT'

/* Update TransAmiga's message pointers */
address command 'fztrump >nil: bbs:configs/message.cfg'
exit 0
```

First action after a session is to tell the mail processor to import all received packets and then export netmail and echomail to the appropriate addresses. After this is done, update TransAmiga's message pointers to make it aware of the new received mails. Note that this command is always executed after a successful session, i.e. a usual fido session or a call by a BBS user. So this command has to serve both tasks.

With TrapDoor (or other Mailers like GMS) you have the possibility to start TransAmiga while the mailer is still running. This is done by the keyword `BBSMODE SPWAN`. Some of you may have restricted memory and running both programs at the same time would cause problems. In this case, you could also start TransAmiga and quit TrapDoor at the same time. This is done by using `BBSMODE EXIT`.

Here are the relevant entries for TrapDoor and GMS:

```
; Entries for TrapDoor to start TA in EXIT mode
BBSCOMMAND "Rx Rexx:StartBBS %B %b"
BBSINOUT ""
BBSMODE EXIT
```

```
; Entries for GMS to start TA in EXIT mode
BBSMode=Exit
BBSCommand="Rx Rexx:StartBBS %B %b"
DropBBSSecs=20
```

Obviously after the user has logged off, you need to start the Mailer again. This is usually done in '`Logoff.trans`'. Here is an example:

```
/* Logoff.trans */
options results
echomail ; e = result
netmail ; n = result
if e = 'TRUE' | n = 'TRUE' then /* check for new netmail or echomail
                            and tell Foozle */
        do
Address 'Foozle' "Rescan CHG"
end
systeminfo '8' ; line = result
if line = 1 then                /* check the line which is running
                                   currently */
do                      /* start TrapDoor again if it is
                                   the normal line */
address command 'c:run mail:bin/trapdoor answer'

                                /* use this line for GMS
                address command 'RUN GMS:bin/GMS_Mailer_GUI line 0' */

end
if line = 0 then
do                      /* kill logfile if this is the local
                                   line */
address command 'delete BBS:system/TransAmiga.log'
end
exit 0
```

Note the check for the line, as this script is always started, also when using the local logon function. When using local mode, you obviously don't want to start TrapDoor after logoff.

Appendix B BBS Tools Reference

This appendix provides a quick reference on the use of all of the included TransAmiga tools. All these programs can be found in Bin directory.

B.1 TransAmiga Full-Screen Editor

Usage: TAEd <*filename*> [-n<*line number*>] [-q<*quote file*>] [-l] [-c]

<*filename*>
: It's the name of the file TAEd shall write the entered text to.

[-n<*line number*>]
: Tells TAEd wich line of TransAmiga BBS it should use.

[-q<*quote file*>]
: Tells TAEd wich file should be loaded into the quote buffer. If a file is specified and TAEd does not find it a error message will be displayd on the used BBS line but work will be continued. In this case or if this option is not used quoting is disabled.

[-l]
: This is a new option. If this is specified TAEd will load the supplied <*filename*> into edit buffer so you can edit existing texts. THIS IS NOT TESTED ACTUALLY.

[-c]
: This new option allows to disable the display of the x-coordinate the cursor has. If you use this option the x-coordinate is no longer displayed in the status lines and the y-coordinate is only updated when the line is really changed.

TAEd Online commands:

CTRL-B
: Jumps to the end of the text. Cursor is positioned after the last character. Not documented in help window.

CTRL-K
: Kill the line the cursor is actually in.

CTRL-L
: Clear the whole text. The edited text is deleted completely. A security question is used.

CTRL-N
: Inserts a line. This seems to work a little weird. I have to examine it a bit more. Seems to be a bug...

CTRL-O
: Quotes the whole text of the quote file. This is done to speed up quoting dramatically. The screen text is refreshed.

`CTRL-Q` Start quotation.

`CTRL-R` Refresh screen.

`CTRL-S` Swiches the display of the x-coordinate in the status line. If `-c` has been used at the command line press this once to activate them. If you press it again the x-coordinate display is turned off.

`CTRL-T` Jumps to top of the text.

`CTRL-U` Deletes text on the left side of the cursor. Text from position 1 to (cursorposition-1) is deleted.

`CTRL-V` Changes editmode between Insert-mode and Overstrike-mode.

`CTRL-X` Deletes the content of a line, not the line itself.

`CTRL-Y` The text is deleted from the actual cursor position to the end of the line. The character the cursor is positioned on is included in deletion.

`CTRL-Z` Saves the whole text to the specified file and exits the editor.

`DEL` Deletes the character the cursor is placed on.

`CR` Ends a line. If cursor is behind the last character of a line a new line will be inserted underneeth the line the cursor is in. If the cursor is somewhere else in the line the text starting under the cursor is taken to the next line and following text is moved one line down - technically a new line is inserted after the actual line and the text is moved to the new line.

`BACKSPACE`
Deletes the character left of the cursor.

`TAB` The cursor is moved one TAB left. There is nothing inserted.

`SHIFT-TAB`
The cursor is moved one TAB back. Nothing is inserted or deleted. It's just moving the cursor faster than cursor keys.

`CURSOR KEYS`
You use the cursor keys to move around in the text. They work normally.

`SHIFTED CURSOR KEYS`
They jump to the start/end of a line/screen. ATTENTION! There are not much terminals that support the codes for this! I just tested it with Terminus 2.0d and it works. Locally it works also. I can not guarantee that it works with other terminal programs.

`ESC ?` Displays the helpwindow.

`ESC ESC` Aborts the message. A security question is in there. If you type `Y` or `Y` or just `<CR>` the message is aborted.

B.2 TransAmiga Access Lock

Usage: TrAL `L|U`

Either locks (L) or unlocks (U) TransAmiga's message areas. When the areas are locked, all message areas become read- only, and users will not be allowed to save messages.

Warning, this only has any real effect while there is a BBS line running (or anything else that uses TransAmiga.library). There is little chance that locking will have any effect if TrAL is used before any BBS lines are running, and there is an equally small chance that the lock will be retained after all lines have been exited.

See Section 5.9 [TransAmiga Access Lock], page 52.

B.3 TransAmiga AminetCD Upload Tool

Usage: TrAminet <*path*>

TrAminet is a tool that processes Aminet CD 4/94. The goal is to produce a filelist for onlineuse with TransAmiga BBS. It checks the file TREE and INDEX on the CD's root directory. Creates the directory stucture of tree on the hard disk and uploads the files listed in INDEX.

You need the AmigaDOS JOIN command. If you did not delete it it should be located in your '`c:`' directory. Also the standard '`RAM:`' is needed. If you don't want to use RAM for that just make a '`RAM:`' assignment to a directory on your hard drive. This is only necessary if you changed the C= default installation to avoid having '`RAM:`'.

AmigaOS 3.1 Assign command or a PD command that supports TRUE multiassign is needed.

You have to create a multiassign first. Let's say that you want to create the needed tree in '`file:CDROM/Aminet`' and your CD-ROM is '`CD0`'. Using the OS3.x-style Assign would look like this:

Assign Aminet: file:CDROM/Aminet Assign Aminet: CD0:Aminet ADD

You have to have this executed during system startup or otherwise TransAmiga wouldn't work correctly. It is important to put the assignment to the writeable device first, because when Ami-

gaDOS searches the logical device 'Aminet:' it will start at the first assign and will only go to the second device when the files it is looking for are not in the first one. Don't switch the assignments or TrAminet won't work (it will try to write to the CD).

Note: AmigaDOS commands DIR and LIST do not know multi-assigns so if you use them to list Aminet: (in my example) you will get the content of the first path of the multi-assign list. In this case it's the directory on the hard drive.

Let me write some words about speed. It takes a long time to process all files so be sure you don't need the shell you start TrAminet in. I don't suggest to use the RUN command as the output to CLI is still there. You can send it to NIL: then there is no argument against run, but you won't get any failtrue messages... Speaking of error messages... There may come up several messages that tell you that JOIN was aborted due to an error. Don't panic, that's ok. The program makes .Desc files out of the .readme files on the CD so the BBS user can read a long description if he wants to. To produce .Desc files in TransAmiga format I have to join two files so I used the join command as it's faster than reading the .readme file and appending it to the .Desc file line by line...

The program produces a fragment for your '`file.cfg`' in '`RAM:`' It is called '`File.cfg`' (what else ?-). The number of found fileareas is displayed in the Shell. (It's 134 for Aminet CD 4/94). Add the number to the number of fileareas in your '`file.cfg`' and insert the '`file.cfg`' fragment in your own '`File.cfg`' where you wish the areas to appear. I suggest at the end, just before the packer definition.

Now you're ready to access the CD from your TransAmiga BBS.

This program was written for AminetCD 4/94 but may work with future releases if they don't change the format. If there is need to update this tool it will be released via the usual means.

B.4 TransAmiga BBS

Usage: TransAmiga [-c<*config file*>] [-b<*baud rate*>] [-l<*locked baud rate*>] [-t<*max time*>] [-n<*line*>] [-x] [-r] [-s] [-0] [-m<*modem connect message*>] [-h] [-p]

Starts up the BBS. See Section 3.1 [Running TransAmiga], page 17.

B.5 TransAmiga BBS Line Activity Display

Usage: TrBLAD [<*line*>]

Displays all current BBS activity in a similar manner to the "Who's Online" command from the main menu. If no specific <*line*> is specified, all lines will be listed.

B.6 TransAmiga Broadcaster

Usage: TrBroad <*message*> -n<*line*>

Sends an online message to the indicated line number.

B.7 TransAmiga Configuration Tool

Usage: TransCfg -c<*config file*>

Setup up and/or edit TransAmiga's configuration. <config file> is the name of TransAmiga's master configuration file.

See Section 2.4 [TransAmiga Configuration Tool], page 10.

List of the possible loglevels that can be defined in the master configuration part:

1	lowest level. Basic and important things are stored. Everything that is not mentioned later.
2	oneliner from user
3	Deleted Message/File
4	Viewed archive/bulletin
5	saved/forwarded msg/moved file
6	password should be/ARexx result codes
7	start/stop of editor/marked message
8	menu commands

9	saving message
10	execute dos cmds
11	waiting for ARexx message

When an up-/download is done and loglevel is ABOVE 5 the Cmd: and Cmd Result: are written to the logfile. Some download counting programs need these lines.

Templates used in the message configuration for the quote lead-in line:

%n	the name of the user who wrote the message you are replying to
%a	the name of the person that message was addressed to
%d	the date the original was written
%t	the time the original was written

Templates used for the external editor startup:

%n	the number of the BBS line starting the editor
%f	the filename of the file that should contain the message text to be saved.
%q	the name of the file containing the quoted text when you are replying to a message.
%b	the real (non-locked) baud rate of the online caller.

Templates used for the protocols in the file configuration:

%n	the number of the BBS line
%d	name of the serial device
%u	serial device unit
%f	the file to be transferred (or directory in the case of a batch transfer)
%l	the name of the file containing the list of files to transfer (created by the BBS for sends)
%b	baud rate of the current caller (the real baud rate, not the locked one, so if you have the serial port locked, don't use this)

B.8 TransAmiga File Lister

Usage: TrFL -o<*out file*> -c<*configs dir*> [-t<*title*>] [-a<*access*>] [-d<*days*>] [-s] [-n]

Creates a readable text file listing files available in the file section.

<*out file*> is the name of the text file to create.

<*configs dir*>
is the name of your TransAmiga configurations directory.

<*title*> is an optional text file that will be inserted at the top of the generated file listing.

<*access*> allows you to limit what areas are included: only areas with this level or lower will be listed in the generated file.

<*days*> defines what is a "new" file. New files are marked with an asterisk (*) in the output file.

`-n` tells TrFL to include only files that are new.

`-s` tells TrFL to sort the output file. TrFL is very intelligent about this, when sorting it will not move around comment lines in the file listing (lines starting with a backslash), and will sort the files inbetween the comment lines, and not move them elsewhere.

B.9 TransAmiga User Maintenance

Usage: TrMaint -c<*config file*> [-n<*line number*>]

Starts the remote SysOp user maintenance module. If you specify a line number, TrMaint assumes that it will be going through the BBS as a door. Otherwise, it will simply run from the CLI.

See Section 3.10 [TransAmiga User Maintenance], page 29.

B.10 TransAmiga Nodelist Look Up Utility

Usage: TrNode <*node number*> <*nodelist path*>

TransAmiga can be setup to work with any nodelist format by simply writing a replacement for TrNode. TransAmiga passes the address to look up along with the nodelist directory to TrNode, eg:

```
TrNode 2:2476/12.0 NodeList:
```

Then TransAmiga looks for the output that TrNode sends to STDIO. The output should look like this:

```
Sami Radwan     (Sysop)
0               (Cost to send message)
Amiga on Phone  (System name)
Karlsruhe       (City it is located in)
```

If an error of any sort occurred, TrNode should set the return code to at least 10, and generate NO OUTPUT.

Remember that TransAmiga uses this program automatically when looking for node information.

B.11 TransAmiga Paragon Door Interpreter

Usage: TrPr <*door path/name*> [-n<*line number*>] [-d]

<*door path/name*>
: is the path/name of the door you wish to start

<*line number*>
: is the number of the BBS line TrPr shall use. The default value for <*line number*> is 0.

`-d`
: enables debug output. This is needed only if you find a door that does not work with TrPr. This slows down the whole thing a lot! Usually it's better to tell me which door is not working and sending the archive of the door to me if it's not available through Aminet.

A little warning: TrPr gets the idle timeout value from TA and returns to the BBS when it encounters the idle time. Be sure your idle setting is high enough as TrPr has no way to cancel the running door. There would be a 'dead' task in your system. Carrier checking is also a responsibility of the door. TrPr follows exactly the Paragon way of informing the door.

Usually you will use TrPr via an ARexx script started from a menu of TA. Here is an example for such a script:

```
/* Example TrPr startup script by Sami Radwan */

OPTIONS RESULTS

SystemInfo '8'
bbsline = result

ADDRESS COMMAND 'BBS:bin/TrPr doors:kr/Knight -n'||bbsline
EXIT 0
```

B.12 TransAmiga Renumber Usrs

Usage: TrRenUsrs <*config file*>

Renumbers the Userfiles. TransAmiga gives each user an additional number and if the user is deleted, these numbers remain unused. If this happens more often, new user numbers get too high values and just irritate. TrRenUsrs just takes the existing userfiles and renumbers them so no gaps are left.

B.13 TransAmiga DOS Shell

Usage: TrShell -n<*line number*> -f<*FIFO name*>

Starts up TransAmiga remote shell. This can only be run from within the BBS as a door.

This is a full interactive shell, supporting things such as simple command line editting, and "raw" mode. It will also check for carrier loss and user idle timeout, in these cases it will send a `CTRL-C` break to the running program.

In order to run, TrShell requires that the '`FIFO:`' device be properly installed on your system. This requires that you copy the fifo.library file to your Libs: directory, the fifo-handler file to your L: directory, which also must be executed in your Startup-Sequence.

For TrShell to work properly, you should first start up a new CLI process, with all it's input and output going through '`FIFO:`'. Note how a unique FIFO name must be generated for each

invocation, in the example below this is done with the ARexx pragma("id") function. An example of starting TrShell from ARexx would be:

```
/* ARexx macro to start the remote shell */
options result
id=time("s")
SYSTEMINFO 8 ; bbsline=result
address command 'NewShell FIFO:trshell'id'/rwkecs'
address command 'TrShell -n'bbsline' -ftrshell'id
exit
```

"trshell" is the unique filename that you must pass to '`FIFO:`' and also to TrShell (they must be identical!); the BBS line number is appended to the end to avoid conflicts on multi- line systems. The "/rwkecs" are the parameters that are required for '`FIFO:`' to work correctly with TrShell.

B.14 TransAmiga Line Status

Usage: TrStat [<*line number*>]

TrStat shows some status information for a given line. If no *line number* is given, TrStat uses 0 as default.

B.15 Tell TransAmiga ARexx Commands

Usage: TrTell [-n<*line number*>] <*command*>

Sends a TransAmiga ARexx command to a currently running line. Usually this is used when you've started TransAmiga with the `-r` option and it is just sitting waiting for an ARexx messages. The same result can be achieved by using the RX command, this way just takes a little less typing.

If your command is more than one word, you should enclose the command in quotes, eg:

TrTell -n0 "baudrate 2400" TrTell -n0 logon

(the commands are case-insensitive).

Sending ARexx messages while a user is online is generally not a good idea, and can lead to unpredictable results. You should only use this command in the situation described above.

B.16 TransAmiga Update File Areas

Usage: TrUFL <*path*>

<*path*> is the path to the directory where you with to upload files to. The files have allready to be placed in that directory. I worked almost two hole days to complete this version so there my be bugs although I tested it really hard.

Special feature is that multi assigns are recognized. .Desc files are created in the same directory the Files.BBS is in. If there is no Files.BBS the first directory of the multi assign list is used. ATTENTION! If a CD-ROM or similar storage system (not accessible for write) contains a Files.BBS in the directory you want to process there will come up a requester telling you that the drive is inaccesible for write operations. To prevent this create a file in a directory of an other device. Connect the directory to the CD-ROM via a multi assign. ATTENTION: The volume where you place the Files.BBS should be the first in the multi assign list. Use Assign from AmigaOS 3.1 or a pd program that allows real multiassigns! Check the doc for TrAminet if you want to know more about multiassign.

IMPORTANT! Never specify a path that leads directly to a writeprotected drive.

B.17 TransAmiga Update Message Pointers

Usage: TrUMP <*msg dir*> [<*msg dir*>...]

This searches for the high and low message pointers in the specified netmail or echomail message directories. This must be run when any program other than TransAmiga adds a message to an area, otherwise TransAmiga will have incorrect values for the message pointers.

Due to slight differences in the way TransAmiga treats local and privileged-only areas, you must never use TrUMP on directories of these types.

B.18 TransAmiga XPR Daemon

TrXPR will now be explained in a little more detail, so that you will be able to further customize your file transfer commands. TrXPR has the following command line usage:

Usage: TrXPR <*line*> <*protocol*> <*init*> R|S <*filelist*>|<*files...*>

<*line*> indicates which BBS line number wants to perform the transfer. You should use `%n` here, so that TransAmiga substitutes the correct line number.

<*protocol*>
is the name of the protocol library to use minus the 'xpr' prefix and the '.library' suffix. So for a protocol that is to use the xprzmodem.library, you would just enter 'zmodem'.

<*init*> is the library initialization string. The meaning of this is dependent on each library. Every library has it's own set of parameters that can be included here, so you will have to consult the library's documentation to find out what to put here.

`R|S` tells where you want to do a Send or a Receive.

<*filelist*> is the name of a file that contains a list of files to be sent.

<*files...*> is the name of the files to be received. How this field is treated depends on the library. With most, it will just be interpreted as the filenames for files to be sent or received. However, some libraries treat it differently. For example, the xprzmodem.library ignores this field when doing a receive.

TrXPR expects the <*filelist*> argument when the S option is used, and <*files...*> when the R option is used. Up to 255 files can be send in one transfer.

TransCfg's defaults are for Xmodem and Zmodem protocols. There are other libraries available, which will allow you to set up Ymodem, Kermit, CIS B+, Jmodem, and other protocols. Using TrXPR, getting these to work with TransAmiga should be quite simple.

This is a part of the '`file.cfg`', setup for some more protocols:

```
 9
Zmodem
 1
BBS:Bin/TrXPR %n zmodem T?,OR,B16,E50,AN,DN,SN s %l
BBS:Bin/TrXPR %n zmodem T?,OT,B16,E50,AN,DN,KY,SN,RN,P%f r ram:
Zmodem special (upto 8k/block)
 1
```

```
BBS:Bin/TrXPR %n szmodem T?,OR,B16,AN,DN,SN s %l
BBS:Bin/TrXPR %n szmodem T?,OR,B16,AN,DN,KY,SN,RN,P%f r ram:
Ymodem-1K
 1
BBS:Bin/TrXPR %n ymodem B1,C1,YG,Z0 s %l
BBS:Bin/TrXPR %n ymodem B1,C1,YG,Z0 r %f
Ymodem
 1
BBS:Bin/TrXPR %n ymodem B1,C1,YB,Z0 s %l
BBS:Bin/TrXPR %n ymodem B1,C1,YB,Z0 r %f
Xmodem CRC
 0
BBS:Bin/TrXPR %n xmodem T0,C1,K0 s %l
BBS:Bin/TrXPR %n xmodem T0,C1,K0 r %f
XModem-1K
 0
BBS:Bin/TrXPR %n xmodem T0,C1,K1 s %l
BBS:Bin/TrXPR %n xmodem T0,C1,K1 r %f
XModem
 0
BBS:Bin/TrXPR %n xmodem T0,C0,K0 s %l
BBS:Bin/TrXPR %n xmodem T0,C0,K0 r %f
Kermit
 0
BBS:Bin/TrXPR %n xprkermit T0 s %l
BBS:Bin/TrXPR %n xprkermit T0 r %f
ASCII
 0
BBS:Bin/TrXPR %n xprascii T0 s %l
BBS:Bin/TrXPR %n xprascii T0 r %f
```

See Section E.3 [File Configuration], page 125 for more information about the format of the '`file.cfg`'.

Appendix C Amiga ANSI Codes

This is a complete references section for the ANSI codes that TransAmiga) understands. For maximum compatibility with various terminal types, TransAmiga understands a combination of both IBM ANSI codes, and Amiga console.device ANSI codes.

C.1 Colour Change Codes

```
ESC [30m       - black foreground
ESC [31m       - red foreground
ESC [32m       - green foreground
ESC [33m       - yellow foreground
ESC [34m       - blue foreground
ESC [35m       - purple foreground
ESC [36m       - cyan foreground
ESC [37m       - white foreground

ESC [40m       - black background
ESC [41m       - red background
ESC [42m       - green background
ESC [43m       - yellow background
ESC [44m       - blue background
ESC [45m       - purple background
ESC [46m       - cyan background
ESC [47m       - white background
```

C.2 Style Controls

```
ESC [0m        - reset ANSI codes
ESC [1m        - boldface on[1]
ESC [3m        - italics on
ESC [4m        - underline on
ESC [7m        - inverse on
```

[1] In 16 colour ANSI, boldface text is rendered as a slightly brighter colour, thus giving you access to the 8 additional colours in the palette.

C.3 Cursor Controls

```
ESC [A          - cursor up
ESC [B          - cursor down
ESC [C          - cursor right
ESC [D          - cursor left

ESC [#A         - cursor up # lines
ESC [#B         - cursor down # lines
ESC [#C         - cursor right # spaces
ESC [#D         - cursor left # spaces

ESC [H          - cursor home
ESC [#H         - cursor to line #
ESC [#;#H       - cursor to a specific row and column

ESC [s          - save the current cursor position
ESC [u          - restore cursor to the last saved position
```

C.4 Miscellaneous Commands

```
ESC [          - insert a space
ESC [#         - insert # spaces

ESC [L          - insert line
ESC [#L         - insert # lines

ESC [M          - delete line
ESC [#M         - delete # lines

ESC [P          - delete character
ESC [#P         - Delete # characters

ESC [K          - delete to end of line
ESC [1K         - delete to start of line
ESC [2K         - delete entire line

ESC [J          - clear to end of screen
ESC [1J         - clear to top of screen
ESC [2J         - clear entire screen
```

NOTE: some ANSI commands can be stacked using the semi-colon (;), eg:

```
ESC [1;43;34m     - blue bold text on yellow background
```

Appendix D Control Sequences

Here is the complete listing of the available control codes.

The uses for these control codes are almost unlimited. They are useful in any textfile, including menu and prompt files, and any other text files the BBS displays.

Most of them are also used by the USERINFO and SYSTEMINFO ARexx commands.

Key	Function
CTRL-A	Prompt user to press RETURN
CTRL-B	Don't allow users to abort viewing
CTRL-C	Enable aborting
CTRL-D	Don't bring up 'More?' prompts
CTRL-E	Enable 'More?' prompts
CTRL-F	Display a piece of user information
CTRL-J	Display 'More?' prompt
CTRL-K	Display a piece of system information
CTRL-U	like CTRL-U without any output (for RIPscrip files)
CTRL-W	Pause for one second

CTRL-F and CTRL-K have special meanings. The character following the CTRL-F or CTRL-K determines what piece of information TransAmiga displays.

CTRL-F combinations are in general for displaying information about the current online caller. They are also used by the ARexx command USERINFO to get the corresponding values:

Key	Function
CTRL-F A	Returns current caller's name
CTRL-F B	User's city, province
CTRL-F C	User's password (careful)
CTRL-F D	User's computer type
CTRL-F E	User's telephone number
CTRL-F F	Date of user's last call
CTRL-F G	User's flags (X is on, - is off)
CTRL-F H	User's chosen language
CTRL-F I	User's protocol
CTRL-F J	User's city
CTRL-F K	User's country/province
CTRL-F L	User's netmail credit
CTRL-F M	Number of messages posted by user
CTRL-F N	User's last read in current conference
CTRL-F O	User's access level

CTRL-F P	Times the user has called the system
CTRL-F Q	Number of files user has uploaded
CTRL-F R	Kilobytes user has uploaded
CTRL-F S	Number of files user has downloaded
CTRL-F T	Kilobytes user has downloaded
CTRL-F U	Total time user has been online today
CTRL-F V	User's screen length
CTRL-F W	User's first name only
CTRL-F X	ANSI status (ON or OFF)
CTRL-F Y	User's screen clearing setting (ON or OFF)
CTRL-F Z	Number of possible downloads before new upload
CTRL-F 0	User's full-screen editor status (ON or OFF)
CTRL-F 1	User's command mode (either Expert or Novice)
CTRL-F 2	User's hotkeys status (ON or OFF)
CTRL-F 3	User's handle
CTRL-F 4	Message read mode (All or Personal)
CTRL-F 5	User's birthdate
CTRL-F 7	User's upload:download ratio
CTRL-F 8	User's street address
CTRL-F 9	User's postal code
CTRL-F !	possible downloads in kbytes before new upload
CTRL-F $	User's kbyte ratio
CTRL-F ?	User's usernumber
CTRL-F %	Name of protocol the user has active
CTRL-F &	Amount of currently marked files
CTRL-F =	User's RIP status (ON or OFF)

CTRL-K combinations return general information about the system. They are also used by the ARexx command USERINFO to get the corresponding values:

Key	Function
CTRL-K A	Number of calls system has received
CTRL-K B	Last caller to system
CTRL-K C	Total number of messages on the system
CTRL-K D	Low message in current conference
CTRL-K E	High message in current conference
CTRL-K F	Sysop's name
CTRL-K G	Name of the BBS
CTRL-K H	High user number on system
CTRL-K I	Current time
CTRL-K J	Current date
CTRL-K K	Loglevel
CTRL-K L	ANSI status (ANSI or ASCII)
CTRL-K M	User's time remaining today
CTRL-K N	Local display status (ON or OFF)
CTRL-K O	Number of available protocols
CTRL-K P	complete connect message

CTRL-K Q	User's daily time limit
CTRL-K R	Baud rate of user
CTRL-K S	kind of logon (LOCAL or REMOTE)
CTRL-K T	LineMsg status (ON or OFF)
CTRL-K U	Carrier check status (ON or OFF)
CTRL-K V	Iconify status (ON or OFF)
CTRL-K W	idle timeout value
CTRL-K Y	Current message conference
CTRL-K Z	Current file library
CTRL-K 0	Number of messages in current conference
CTRL-K 1	Number of current conference
CTRL-K 2	Number of current library
CTRL-K 3	Current one-liner
CTRL-K 4	Who the current one-liner is from
CTRL-K 5	Commands available from current menu (command key of each command, separated by spaces)
CTRL-K 6	Path to current message conference
CTRL-K 7	Path to current file library
CTRL-K 8	Number associated with this BBS line
CTRL-K 9	Master configuration file this line is using.

Appendix E Technical Details

This is intended to those developping tools and utilities for use TransAmiga v1.2. It provides the various specifications necessary.

E.1 Master Configuration

This file can actually be called anything, and can be located anywhere. The path and name will be stored in an ENV-variable on startup: '`ENV:TA.<`*line number*`>`'. It is an ASCII text file, even numeric values are stored in ASCII representations. Each line of the file contains a single piece of data. Here is the line by line format:

```
 1: System path
 2: Configuration files path
 3: User files path
 4: Text path
 5: Bulletin path
 6: Extras path
 7: ARexx path
 8: Maximum baud rate
 9: Modem initialization string
10: Modem exit string
11: Modem local logon string
12: Modem local logoff string
13: Name of serial device
14: Serial device unit
15: New user time limit
16: New user ratio
17: New user access (Default: 10)
18: New user flags
19: Privileged access
20: Idle time length (seconds)
21: Name of the BBS
22: SysOp name
23: Screen type (0=Workbench, 2=Custom, 4=Interlaced custom)
24: Number of bitplanes for custom screen
25: Open blue backdrop window (0=No, 1=Yes)
26: Custom screen width (0=WB screen width)
27: Custom screen height (0=WB screen height)
28: X co-ordinate of top left corner of terminal window
29: Y co-ordinate of top left corner of terminal window
30: Terminal window width
31: Terminal window height
```

```
32: Name of font to use for terminal window
33: Point size of font
34: Input colour number
35: Prompt colour number
36: Trim colour number
37: Minimum baud rate
38: Sizing gadget on terminal window (0=No, 1=Yes)
39: Prompt field for ANSI callers (0=No, 1=Yes)
40: Guest user time limit
41: Guest user ratio
42: Modem answer string
43: Loglevel
44: New user byteratio (kbytes, max: 2147483647 - 0 disables)
45: Guest user byteratio (kbytes, max: 2147483647 - 0 disables)
46: Task priority of TransAmiga. (-3 to 2 should not be exceeded
                                  default is 0)
47: Stop onlinetime decrease while chat (0=decrease, 1=stop it)
48: Turn notification of other lines on/off when a user logs on.
    0 or blank line turns notification off, 1 turns it on.
49: If  you  place  a 1 in this line domains are inserted to the origin
    line of fidonet  mails.  If you don't want them to be inserted place
    a 0 here.  The default action is to insert the domain.  You can
    disable domains for single nets  by  leaving the corresponding line of
    message.cfg blank.  The default value  is used if there is no entry in
    that config line - TA just acts like it did till now.
50-54: Blank lines for padding
```

E.2 Message Configuration

This file is always called '`Message.Cfg`' and kept in the configuration directory. It follows the same ASCII format as other configuration files. Here is the format, line by line:

```
 1: Number of FidoNet addresses
    For each address:
     1: Zone
     2: Net
     3: Node
     4: Point
     5: Pointnet
     6: Domain
 2: Inbound directory
 3: Outbound directory
 4: Nodelist directory
 5: User name that feedback gets sent to
 6: Conference number to put feedback in
 7: Remote ANSI full-screen editor command
 8: Local editor command
 9: Allow use of handles (0=No, 1=Yes)
10: Automatically switch to next message conference (0=No, 1=Yes)
11, 12: Blank lines for padding
13: Number of message conferences
    For each conference:
     1: Name of conference
     2: Path to conference
     3: Conference type (0=Local, 1=Privileged, 2=Matrix, 3=Echo)
     4: Allow handles in this conference (0=No, 1=Yes)
     5: Read access
     6: Write access
     7: Conference access
     8: Conference flags
     9: Origin line (left blank for non-echo type areas)
    10: Quote lead-in line
    11: FidoNet address number to use
    12: Blank line for padding
```

E.3 File Configuration

This file is always called '`File.Cfg`' and lives in the configuration directory. It follows the same type of ASCII format as the other configuration files. Here is the format, line by line:

```
1: Minimum number of free kilobytes to allow uploads
2: Maximum number of marked files
3: Blank line for padding
4: Number of file libraries
   For each library:
    1: Library name
    2: Path to library
    3: Read access
    4: Write access
    5: Library access
    6: Library flags
    7, 8: Blank lines for padding
5: Number of archiving methods to support
   For each method:
    1: Archiver name
    2: Archiver suffix
    3: View archive command
6: Number of transfer protocols to support
   For each protocol:
    1: Protocol name
    2: Protocol type (0=Normal, 1=Batch)
    3: Send file command
    4: Receive file command
7: Number of archiving methods to support File_ID.DIZ
   For each method:
    1: Archiver suffix (e.g. .LHA)
    2: Extract command (%f inserts the path and file, %w inserts the
                        target directory ( T: ) Example:
                        LHA e %f File_ID.DIZ %w
                       )
```

This file is always called '`Bulletin.Cfg`' and lives in the configuration directory. It follows the same type of ASCII format as the other configuration files. Here is the format, line by line:

```
1: Number of bulletins
   For each bulletin:
   1: File name of bulletin (always in Bulletins directory)
```

If the file '`NoNotif.Cfg`' is present in the configs directory of the line, it is scanned. Each line of the file has to contain a single username.

ATTENTION! Spacing between the parts of the name is important!

If someone logs on who is listed in this file other lines are NOT informed.

Useful for the sysop if he wants to log on locally while other users are online and he doesn't want them to know that he is present.

E.4 User Files

TransAmiga stores user files in the Users directory as specified when the user sets up with TransCfg. Each user that joins the BBS gets their own seperate file, which is placed in this directory with a filename exactly the same as their user name on the BBS. Due to this method, special care must be taken which characters are used in the user's name, so TransAmiga only allows chr(32)-chr(126) (but no : or /) for user names and handles. As with almost all files used by TransAmiga, this is in an ASCII text format, even numerical values, with one piece of data per line. Here is the format, line by line:

```
 1: The user's street
 2: City
 3: Province (or state, or territory, whichever)
 4: Phone number
 5: Computer type
 6: Birthdate (YYMMDD)
 7: Postal code
 8: Handle
 9: Date of last call (YYMMDD)
10: Password (10 chars. max.)
11: Total kilobytes downloaded
12: Total number of files downloaded
13: Total kilobytes uploaded
14: Total number of files uploaded
15: Messages posted
16: Number of calls to the system
17: Screen length
18: File ratio
19: Daily time limit
20: Time unused today
21: Command entry mode (0=command-stacking, 1=hot-keys)
22: Graphics mode (0=ASCII, 1=ANSI)
23: Editor type (0=line, 1=full-screen)
24: Access level
25: Netmail credit
26: Flags (see below for how flags are stored)
27: Help level (1=novice, 0=expert)
28: User number (do not alter)
29: Language the user chose.
30: Transfer protocol
31: Screen clearing active (1=Clearcodes, 0=none)
```

```
32: Bytefileratio in kbytes
33: Sysop's comment on user
34: RIP mode (0=OFF, 1=ON) May be combined with line 22 in future
    so don't use this line directly!
35-45: Blank lines for padding.
```

The user directory also contains a file called '`Users.BBS`', which is the high user number. This is kept track of so that new users can be given a unique user number. Never alter this file.

If a user has a handle, then a file will also appear in the Users directory with the handle as the filename, but prepended by an . This file contains one single line holding the user's real name.

E.5 File Listing

For each file area setup, TransAmiga keeps a file called '`Files.BBS`', which is also a text file, and contains the name, size, upload date, and description, of each file in that area. The format of this file is to have one file entry per line, with the following fields. Gaps are padded with spaces.

filename - 23 characters
a space
file size - 7 characters
a space
upload date (YYMMDD) - 6 characters
a space
file short description - 40 characters or less

Additionally, each file in the file area should have an accompanying '`<`*filename*`>.Desc`' file, which contains the uploader's name on the first line, with each additional line being the long-description.

E.6 Message Pointers

TransAmiga requires two files to hold message pointers for each message area. One for the high and low message numbers ('`Message.BBS`') and one for user last read pointers ('`LastRead.BBS`'). '`Message.BBS`' merely contains the ASCII values of the low and high message number for a FidoNet area, and for local areas, it stores the low and high message numbers, and the message limit in that area. In both files, each value appears on consecutive lines.

The last read pointers file is a little different than most TransAmiga data files, in that the values are storred in binary form. It is a simple relative file, with one record per user of the BBS. Each record is made up of a signed 16-bit integer, which represents the users' last read message in that area, or a value of -1 if the user has that area masked out. Which record a user's last read pointer goes in is determined by the user's user number (this is if you are referring to the first record as number 1 and not 0).

The files are only read when necessary, e.g. when the conference is changed or new messages are searched.Lastread pointer are updated when changing message conferences using `>`, `<`, or `RETURN`. Thus during the logoff or relogin procedure only the lastread pointer of the actual conference is saved.

E.7 Miscellaneous Informations

Flags in Data Files:

Instead of storing flags in an interpreted form (eg. X–XX–X) in data files, TransAmiga stores them in a packed integer form. Each flag is considered to be a bit, with bit 0 starting at the left, and ending in bit 7 at the right. An X is treated as "True" and a - as "False". Therefore, TransAmiga will store X–XX-X- as 1+0+0+8+16+0+64+0=89.

E.8 Runtime Error Codes

As runtime errors occur from time to time and are not catched completely I list the verbouse meaning of them:

```
3  : RETURN without GOSUB
4  : Out of data
5  : Illegal function call
6  : Overflow
7  : Out of memory
9  : Subscript out of range
10 : Redimensioned array
11 : Division by zero
13 : Type mismatch
16 : String formula too complex
20 : Resume without error
31 : Wrong number of subscripts
49 : Volume not found
```

```
50 : Field overflow
51 : Internal error
52 : Bad file number
53 : File not found
54 : Bad file mode
55 : File already open
57 : Device I/O error
61 : Disk full
62 : Input past end
63 : Bad record number
64 : Bad file name
67 : Too many files
68 : Device function unavailable
70 : Disk is write protected
75 : Path/file access error
76 : Path not found
77 : Break pressed
```

That are all codes listed in the manual. When a runtime error occurs a requester tells you the number and where it failed. Most runtime errors come up when something with the configuration is wrong, e.g. a file is too short or missing. When there is a problem use SnoopDOS or a similar program to find the file that caused the trouble.

Appendix F Known Problems

This is a list of problems that may occur when using TransAmiga. Most of them are problems related to HiSoft Basic or some obscure programming structures in TA which were introduced by the original programmer and cannot be changed on the fly. This part of the documents will surely vanish in the TA2 docs, at last we hope so :-)

- General Problems
  - On some configurations, problems with the BBS shell may occur due to some bugs in FIFO.
  - Not all DOS Doors using standard I/O will work with TrShell. This is because TrShell uses a RAW shell and FIFO, which doesn't recognize all input events. Actually, the door works, but the user inputs won't reach the door. An example is DeepSpace.
  - Opening TA as a window on the Workbench only allows screen dimensions upto 640x400. This is a restriction in HiSoft Basic.
  - Don't forget to put '`BBS:Bin/`' (or whereever your TA executables are) into the system path, or '`TrNode`' cannot be executed (becaus it could not be found).
  - The speed of the local display may vary depending on the version of the '`TransAmiga.library`' when you do not use a graphics card. This is usually only recognizable when the display is scrolling.
  - On public screens ,the ANSI background color *Dark Blue* is used incorrect (HiSoft problem with 2.0+ screens) and the window title gets a wrong background color when the window is active.
  - Sometimes after the occurence a second Runtime Error in one run, TA will crash or shut down the screen. This behavior is caused by HiSoft-Basic and cannot be predicted. It is not the fault of TA.
- ARexx-related Problems
  - Never start 2 ARexx scripts on the same line! This can happen quite easily when a script is already running and you use the chat option (having a chat script installed) or using the custom menu.
  - A basic thing in self written scripts: never forget to use `options results` or your scripts won't work correctly.
  - When starting the Editor from an ARexx script, '`editor.trans`' isn't started.
- File-Transfer Problems
  - When having problems with batched downloads of more than 9 files, an older version of the '`TransAmiga.library`' is used. Get a newer one and reduce batch download to a maximum of 9 files per batch in the meantime.

– XPR libraries should not be packed with XPK, Imploader, Powerpacker or any other online packer or they usually do not work.

Appendix G Glossary

`Access Levels`
: A means of restricting users from using certain parts of the BBS. When an access level is specified for a command or area in TransAmiga, that function becomes invisible to users who have an access level lower than that number. Valid access levels range from 1 to 255. Additionally 8 access flags can be defined, so you can create up to 65536 different user access levels in theory.

`ANSI Graphics`
: A format for sending simple graphics information. It has codes for changing colours, moving the cursor, etc. Most terminal programs and BBS's support this protocol.

`Archive`
: A single file that contains several other files encoded into it or a complete image of a floppy disk. Usually some kind of compression is done on these files or disks. Special tools are needed to add and extract files/disks from archives. There exist various archiving methods such as: LHA and Zoo, which are characterized by the suffixes .LHA and .zoo on the filenames of their respective archives, for files amd DMS or ZOOM, characterized by the suffixes .DMS and .ZOM, for disks.

`ARexx`
: A high level scripting language for the Amiga that is unique in that it defines a protocol for inter-process communication, allowing non-programmers to create macros that control other software. Part of AmigaOS since v2.x.

`Batch Transfer`
: A transfer in which several files will be sent automatically one after another where no extra work is required by the users between each transfer. Only certain protocols such as Zmodem are capable of batch transfers.

`BBS`
: Bulletin Board System. A piece of computer software that will automatically receive calls over the modem and allow callers to perform various functions such as posting and reading messages, sending and receiving file, etc.

`Door`
: A program that is executed by the BBS to perform some sort of outside function. Now an obsolete term, as most BBS packages work interactively with external programs these days.

`Download`
: To receive a file over the modem.

`Echo`
: A FidoNet message area, where all messages posted are sent through FidoNet to all other systems that wish to receive that echo. In essence, a public message conference.

`FidoNet`
: A non-profit publicly accessible network of several thousand computer users all around the world (but predominantly in North America, Europe and Australia) that allows for the transferring of messages and files. Currently, over 20000 systems are listed in FidoNet.

`File_ID.DIZ`
: A standardized text file included in an archive to describe its contents. As the file has always the name '`File_ID.DIZ`', it's easy to detect and extract.

`Freeware`
: A concept of software distribution in which the author(s) retain a copyright over the program, but require no fee to use it.

`FTN`
: Abbreviation used for nets using FidoNet technology without necessarily being part of FidoNet (FTN = Fido Technology Network). Usually, they use other zone numbers than FidoNet (1-6 at the moment), e.g. AmigaNet uses 39-41 as zone numbers.

`FTS-0001`
: The basic FidoNet technical document that describes the message format, packet format, and mailer protocol.

`Kludge Lines`
: Special lines that appear in FidoNet messages that are intended to give more information to programs processing FidoNet messages. A CTRL a (ASCII code 1) precedes them to identify them as kludges. TransAmiga displays these in bold text and can only be viewed by those with privileged access.

`Mailer`
: A program that makes and receives calls for the delivering and picking up of mail bundles and other files. Used within FidoNet. Usually this will also act as a frontend to a FidoNet BBS, answering incoming calls and spawning the BBS when necessary. Eg. TrapDoor, JamMail.

`Mail Processor`
: A program used to unpack incoming mail bundles and toss them into the correct areas, and to pack up outgoing mail into bundles ready for sending. Eg. Foozle, TrapToss, MailManager.

`Matrix Mail`
: See Netmail.

`Message Header`
: The part of the message that contains such information as who sent it, who it was sent to, date it was sent, etc.

`Multi Assigns`
: Multi Assigns make it possible to define a logical device to be spread over several devices, either logical or physical. Available since AmigaOS3.1 within the OS itself and by the use of some multiassign tools. Multi Assigns are used by TrAminet to create the '`Files.BBS`' and '`.Desc`' files for a write protected medium (CD-ROM) which otherwise couldn't be used via TransAmiga's '`File.Cfg`'.

`Multi-Line`
: The ability for a BBS to somehow handle multiple callers on multiple phone lines at the same time.

`Netmail` Messages sent through FidoNet to a specific person at a specific FidoNet address. Usually used to send mail that is intended to be relatively private. Also referred to as Matrix Mail.

`Path Name` Refers to the full AmigaDOS reference to the location of a file. This includes the device or volume name which should be followed by a colon (:); followed by any directories that the file may be in, followed by a slash (/); followed by the file name itself. Example: Sys:Utilities/Calculator

`Public Domain`
When referring to software, it means that the author(s) gives up all rights over the software, allowing anyone to do whatever they like to it.

`Protocol` A special means to transfer files over the modem that insures that the file arrives error free. Examples are Xmodem, Zmodem, Xmodem-1k, etc.

`Return Code`
A code sent to AmigaDOS by a program when it exits. Usually to provide information about what that program has done so certain actions can be handled by a script file.

`Session Handler`
See Mailer.

`Shared Library`
A collection of predefined functions that programmers can use in their programmers. They can either reside in the Amiga's ROM (such as the intuition.library) or on disk in the Libs: directory (such as the diskfont.library).

`Shareware`
A concept of software distribution where some form of the software is provided free, and users are required to "register" after a certain period of time. Registering normally includes a fee.

`Upload` To send a file to another system over the modem.

`Upload:Download Ratio`
A means of restricting people from downloading too much, and forcing them to upload. For every specified amount of files they download, they must upload one. If the reach the download limit without a corresponding upload, they will not be allowed to download anymore until a file is uploaded. Ideally, a circle is created: the more people download, the more they must upload, and the more files there are for downloading, the more people will download.

`XPR Protocols`
A method of implementing file transfer protocols externally on the Amiga. Each protocol lives in its own shared library, which contains a standard set of functions for transferring files. An XPR supporting application can be made to work with any number of protocols simply by installing new libraries.

10 Index

Text and Config File Index

T

ARexx Commands Index

M

N

P

Q

R

S

T

U

V

W

Y

Program Index

Concept Index

Q

R

S

T

U

X

Table of Contents